LEARNING PROBLEMS IN TRANSPORTATION NETWORK

Sankarshan Mridha

LEARNING PROBLEMS IN TRANSPORTATION NETWORK

Thesis submitted to the
Indian Institute of Technology, Kharagpur
For award of the degree

of

Master of Science

by

Sankarshan Mridha

Under the supervision of

Dr. Sourangshu Bhattacharya
Dr. Niloy Ganguly

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

May 2017

APPROVAL OF THE VIVA-VOCE BOARD

Date: / / 20

Certified that the thesis entitled **"Learning Problems in Transportation Network"** submitted by Sankarshan Mridha to the Indian Institute of Technology, Kharagpur, for the award of the degree of Master of Science has been accepted by the external examiners and that the student has successfully defended the thesis in the viva-voce examination held today.

(Member of DAC) (Member of DAC) (Member of DAC)

(Supervisor) (Joint Supervisor)

(External Examiner) (Chairman)

CERTIFICATE

This is to certify that the thesis entitled ***'Learning Problems in Transportation Network"****, submitted by Sankarshan Mridha to the Indian Institute of Technology, Kharagpur, for the partial fulfillment of the award of the degree of Master of Science in Computer Science and Engineering, is a record of bona fide research work carried out by him under my supervision and guidance.*

The thesis in my opinion, is worthy of consideration for the award of the degree of Master of Science in accordance with the regulations of the Institute. To the best of my knowledge, the results embodied in this thesis have not been submitted to any other University or Institute for the award of any other Degree or Diploma.

Sourangshu Bhattacharya
Assistant Professor
Department of Computer
Science and Engineering,
IIT Kharagpur

Niloy Ganguly
Professor
Department of Computer
Science and Engineering,
IIT Kharagpur

Date:

Date:

DECLARATION

I certify that

a. the work contained in this thesis is original and has been done by me under the guidance of my supervisors.

b. the work has not been submitted to any other Institute for any degree or diploma.

c. I have followed the guidelines provided by the Institute in preparing the thesis.

d. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

e. whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Sankarshan Mridha

ACKNOWLEDGMENTS

ABSTRACT

Today's urban lifestyle mandates reliable traffic estimates, allowing citizens to make productive use of their time. However, achieving accurate traffic estimates is rendered difficult, in the presence of unprecedented traffic situations and sudden disruptions on routes. Existing techniques for travel time estimation either rely on data collected from loop detectors or from GPS traces of 'online' users; while the former strategy is expensive, the latter is infeasible in regions with poor connectivity. In this thesis, our goal is to utilize publicly available crowdsourced datasets, with limited information, to accurately perform traffic estimates. Our solutions include providing robust travel time estimates, as well as identifying traffic trend-shifts in response to sudden disruptions.

The first task undertaken in this thesis is travel time estimation supplemented with reliability measures for routes in the road network. A route is deemed reliable when its travel time variation is low – this is desirable for many travellers who do not mind taking a slightly longer route (than the shortest one) but are guaranteed to reach within a stipulated time duration. In a departure from existing works, we utilize public crowdsourced datasets, carrying limited information, and perform extensive data engineering to accurately estimate travel time.

In the subsequent task, we attempt to learn how dynamic traffic demands on one road link affect traffic demands in other links, thereby impacting travel time heavily. Recent works assume that the travel time of a link is spatially correlated only with its neighbouring links within a local area, which is not the case always. Instead, we propose a method which does away with such proximity constraints, and explore the correlations of travel times on different links of an area.

In our final task, we strive to analyze the effects of sudden closure of road-links, due to events such as parades, carnivals, road maintenance activities, etc. The resulting deviation from regular traffic patterns manifests in change of pickup locations for different public transport machineries, such as the *yellow taxi*, causing inconvenience to drivers and passengers alike. Such occurrences motivate us to predict the new pickup locations due to disruption of a road-link, thereby benefitting both parties.

In summary, this thesis endeavours to accurately estimate travel durations and their correlations, from publicly-available crowdsourced data, and to predict geographical shifts in passenger pickup locations in response to link disruptions.

Keywords: Spatio-temporal data mining, Travel time estimation, Path certainty estimation, Route recommendation, Taxi pickup hotspot prediction, Hotspot relocation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"You can't understand a city without
using its public transportation system."
— Erol Ozan, *Talus*

Public transport is the lifeline of a modern urban society: a healthy public transport system demands that citizens be accurately and dynamically informed about travel durations and changes in transport dynamics. Real-time travel information enables citizens to undertake well-planned, hassle-free journeys, allowing them to make productive use of their valuable time. However, accurate travel estimates are difficult to achieve in a rapidly evolving traffic system, where traffic demands are incessantly changing and road link disruptions are sudden. Existing strategies for travel time estimation and traffic dynamics prediction either rely on data collected using advanced machineries such as loop detectors and probe vehicles, or GPS traces aggregated from the cellular phones of 'online' users. However, both these strategies suffer from their fair share of limitations: while the former strategy is expensive, the latter is impractical in zones with poor internet connectivity. In this thesis, we leverage on publicly available crowdsourced datasets to estimate travel durations. Furthermore, we observe that local events such as carnivals, road-shows, prides, rallies, construction works, etc. often lead to sudden closures of road links, which lead to geographical shifts of pickup points for public transport. Employing extensive data engineering techniques on the available datasets, we predict these shifts in advance, such that

consumers and service providers are equally benefited.

1.1 Motivation

For any transportation network, road intersections and road links (segments) are the key observation points. Because the behaviour of traffic depends on how the congestion flows in the links, how the traffic gets delayed in the road intersections, how the traffic in one link affects the other links and so on. Therefore, in this work one primary focus is to study these links and their correlations for a better understanding of travel time distribution in the road network. Now estimation of link travel time [12, 44, 49, 55] is a core problem in transportation engineering, with applications ranging from route recommendation for the individual users [3, 22, 28], detection of traffic behaviour and anomalies at different places and times [2, 10, 35], assessing and improving overall efficiency of transportation systems [44], etc. Further, to understand the traffic pattern, congestion behaviour and their correlation with other external factors like weather condition, the transportation network has also been augmented with other type of heterogeneous information source like online social network [38, 52], weather data, humidity sensors [57, 60] etc. Useful urban traffic related posts from Facebook [38], spatio-temporal tweet feed from Twitter [52] have been used to predict traffic related incidents. Now estimation of link travel time often leads to the search of efficient route finding. Currently Google Maps provides excellent route navigation system, but relies on users being *online*, for collection of traffic data. This is not sustainable in many developing countries, since Internet connectivity is expensive and not as ubiquitous as the developed countries, leading to google maps announcing offline mode[1]. However, in recent times, many of these places are getting partially connected with many online taxi hire companies, such as Uber[2], Ola[3] etc., which store at least pickup and dropoff locations of each taxi trips. Estimating link travel times, learning link correlations, understanding traffic behaviour around a closed link from such trip end-point data can potentially lead to a more sustainable travel time estimation technique and traffic behaviour analysis for connectivity-poor regions.

[1] http://www.digitaltrends.com/mobile/google-maps-youtube-offline-news/

[2] https://www.uber.com/

[3] https://www.olacabs.com/

1.1.1 Need for understanding transportation network from crowdsourced data

Traffic in the transportation network heavily influences the travel time from origin to destination. Existing systems for travel time estimation either use data collected from loop detectors [3, 26] and probe vehicle locations [41, 64], or from GPS traces from cellphones of online users (e.g. GoogleMaps[4]). The former methods of data acquisition are expensive, while the latter turn out to be infeasible in connectivity-poor regions. But in recent times, several cities around the world have started online taxi hire companies, such as Uber, Ola, etc., which collect route information. Besides, in many cities, the city administration makes it mandatory for taxi hiring services to log their trip information. These taxi service organizations are increasingly making their taxi trip data public (e.g., Uber data[5]). However, understandably due to various constraints such as privacy concerns of the driver and the passenger, legal issues, business interests, etc., hardly any dataset contains complete trajectory information. For example, sometimes trip information is anonymized, or only pickup information is provided, or no distance information is shared, or exact pickup-dropoff information is masked. But despite containing limited information, they are sufficient for inferring meaningful insights after thorough data engineering. The datasets are both cheap to acquire (hence available in large volumes), and impose less heavy connectivity requirements on the end users.

In many countries, the city authorities have made the transportation data public. First is *Porto Taxi Dataset*[6] which contains intermediate trajectory information at a regular interval of 15 seconds for the taxi trip in Porto City. The data confirms only the pickup information and doesn't share the total duration for each trip [32]. Second is *Boston Taxi Dataset*[7]. This seven months data contains only pickup, dropoff co-ordinates and timestamps but no other information. Then there is *Bejing Taxi Dataset*[8], *Rome taxi dataset*[9], *SFMTA data*[10] from

[4] https://maps.google.com/
[5] https://movement.uber.com/cities
[6] http://www.geolink.pt/ecmlpkdd2015-challenge/dataset.html
[7] https://data.cityofboston.gov/Transportation/Boston-Taxi-Data/ypqb-henq
[8] http://sensor.ee.tsinghua.edu.cn/datasets.html
[9] http://crawdad.org/roma/taxi/20140717/
[10] ftp://avl-data.sfmta.com/avl_data/avl_raw

San Francisco Municipal Transportation Agency (SFMTA) and *San Francisco taxi data*[11]. These datasets contain taxi coordinates at regular interval of time along with timestamps. However, to maintain privacy, they never specify which of the coordinates are pickup and/or dropoff. Thus it provides only the taxi footprint but no trip level information. Then there is Bike trip Data from *Washington DC*[12], *New York City*[13], *Chattanooga*[14] which contains only the pickup and dropoff information and total duration. But no intermediate trajectory information and trip distance are provided. Finally there is *NYC Taxi data* from New York city Taxi and Limousine Commission which shares only the pickup and dropoff information but no intermediate trajectory details. But they share the total trip distance and duration for each year from 2009. So, there is a large pool of publicly available transportation data, which if utilized properly can be very economically feasible for this kind of study in the developing countries. This motivates us towards building some framework for utilizing these incomplete data and then learn various transportation network problems for inferring meaningful insights.

1.1.2 Reliability of road links while travelling

The problem of finding the shortest paths in the transportation network has been heavily investigated [4, 16, 18, 65] due to its wide application in transportation science and other fields (e.g., route guidance systems, traffic simulations, or logistics optimizations). But the traffic networks are highly uncertain due to fluctuations in demand and inconsistency in supply [46]. This results in unstable link travel time estimation [37]. Also it's not straightforward that a shortest distance path will always require least amount of time to travel. On the contrary, it's very intuitive that if multiple paths exist between an origin destination pair OD, then there are high chances that most people will follow the shortest distance path – resulting higher congestion, hence higher fluctuation in travel time. So if a person has a hard deadline (e.g. to catch a flight, train etc.) to meet, then she should either avoid paths with high fluctuation or start early considering worst case scenario. A relatively longer distance path may invite lesser traffic and thus may have lesser fluctuation in travel time. Conse-

[11] http://crawdad.org/epfl/mobility/20090224/
[12] https://www.capitalbikeshare.com/system-data
[13] https://www.citibikenyc.com/system-data
[14] http://bit.ly/2kTb7tJ

quently, its worst case behaviour may be better than that of the shortest path. Therefore, in order to develop a route planner, besides having an estimation of average travel time, assessment of its variation is also a necessity. This motivates us to investigate the variation in travel time of constituent links of the path apart from understanding the average link travel time during different hours of the day.

1.1.3 Correlation of road links during traffic

The spread of road traffic in urban area follows a complex pattern. Congestion in one link may quickly unsettle nearby or sometime distant regions – resulting in inconvenience to the city people. Therefore an interesting problem is to investigate the geographical range of influence of traffic occurring at a transportation network. More precisely, the motivation is to understand how the link behaves – their correlation with each other during traffic. Understanding the travel time correlation between the links has its own importance. It helps to detect anomalies in traffic behaviour in a spatio-temporal setting [35] and thus improving the overall efficiency of the transportation system [44]. Till now the study of the interdependence of link travel time assumes that the links effect only the nearby links [33, 34] or they are restricted to a local area [11]. However, it is often found that in city area long range correlations exist. That is, congestion in one part of the city adversely affect another remote locality - identification of this may be important for various city planning related applications. Therefore, a modelling paradigm which will not be restricted to the locality of the correlation assumption is a necessity.

1.1.4 Effect of disruption in transportation network

The behaviour of traffic changes dramatically if a thoroughfare is closed. Within a city, an experienced taxi driver can envisage the probable passenger pickup locations and subsequently from collective experience, taxi pickup hotspot emerge. The city residents also identify these locations ensuring an organic reinforcement. The hotspots in turn induces certain trajectories which the drivers follow regularly. However, the trajectories and consequently the hot spots may get shifted if there is any disruption on a road segment due to any

natural (e.g. snowfall, heavy shower etc.) or man made (e.g. parade, protest march etc.) event. Sometimes a regular hotspot may lose its existence and a new hotspot may appear into the city. Also these new hotspots may be far away from the regular ones – causing trouble to both the passengers and the drivers as they may not have prior information regarding this change. This motivates to investigate this interesting problem of predicting the change in hotspots, which if known, would be useful for both the passengers and the taxi drivers.

1.2 Objectives of the thesis

The objectives of the thesis are to learn how travel time is distributed over individual links, their correlations with each other and geographical change in traffic trends during sudden link disruptions. More specifically, we tackle the following three aspects.

- How path travel time is distributed over intermediate links: Travel time is heavily influenced by the imbalanced traffic congestion in the road network. This results in the non uniform distribution of the path travel time over the constituent links. The primary objective is to estimate the average link travel time and its variance. Followed by which we try to recommend the path with maximum reliability or minimum variance in travel time.
- How link travel time is correlated with each other: Due to fluctuations in demand and supply in different road links, the congestion in one link may influence (positively or negatively) the congestion in other links. Therefore, the next objective of the thesis is to analyze the correlation of road links while learning the travel time distribution of the links in the road network. We further explore the long range link correlations without putting any proximity constraints.
- How traffic pattern changes geographically during link disruptions: Our final objective of the thesis is to learn the geographical change in traffic pattern when a particular road link is closed due to various reasons like parade, carnival, road maintenance etc. Here we investigate how the popular passenger pickup locations for public transport like yellow taxi are changed due to disturbance in traffic pattern due to link disruptions.

1.3 Contributions of the thesis

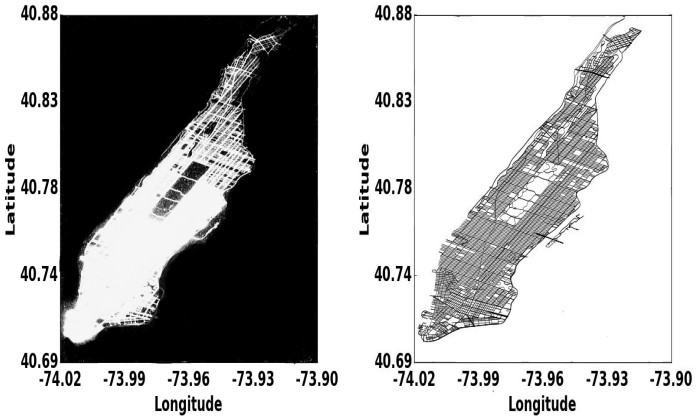
The primary focus of the thesis is to devise a framework to investigate the transportation network using crowdsourced traffic data. The thesis focusses on estimatation of average link travel time, the correlation in travel time among various links and also the geographical change in traffic patten during sudden link disruptions.

The three salient contributions of the thesis are as follows:

- Link travel time prediction and route recommendation.
- Estimation of covariance in link travel time.
- Prediction of taxi pickup hotspots during road closure activities.

1.3.1 Link travel time prediction and route recommendation

Existing systems for travel time estimation either use sophisticated machineries like loop detectors and probe vehicles, or from GPS traces from cellphones of "online" users. The former methods of data acquisition are expensive, while the latter turn out to be infeasible in regions with poor connectivity. However, many crowdsourced taxi trip datasets (from Boston, Beijing, Rome, etc.) are publicly available which, despite containing limited information, are sufficient for inferring meaningful traffic insights after thorough data engineering. The datasets are both cheap to acquire (hence available in large volumes), and impose less heavy connectivity requirements on the end user. One such crowdsourced dataset is the NYC (New York City) Taxi dataset, which contains only the end-point (pickup and dropoff) information for each trip. Figure 1.1(a) shows the taxi pickup, dropoff density (white part) in Manhattan city and figure 1.1(b) shows the corresponding road network. In this work, we develop a link travel time estimation algorithm from such end-point data. The key idea is to augment a subset of trips with unique paths using logged distance information, as opposed to fitting adhoc "route-choice" models. Also a route recommendation system is built using the estimated link travel times and is compared with existing commercial and state-of-the-art systems with encouraging results.



(a) (b)

Figure 1.1: (a) Original pickup, dropoff distribution (white part) across Manhattan (b) Constructed full road network for Manhattan

Link Travel Time Estimation

Let $G = (V, E)$ denote the road network, where V is the set of all *nodes* which are the origin and destination points of various trips, $E \subseteq V \times V$ is the set of all road *links* in the network. We assume a directed graph with $e = (v_1, v_2)$ implying that traffic can flow from v_1 to v_2. A path p is an ordered list of nodes (V_p) such that $(v_{i-1}, v_i) \in E, \forall i$. The Manhattan ground truth trips dataset $\mathcal{D}$ can be described as a collection of paths p_i along with trip travel time $T_i, \forall i = 1, \ldots, N$, where N is the total number of trips. Note that a given path p_i may have multiple trips. Hence, we can compute the mean travel time $\bar{T}_p$ and variance S_p for a path p.

In order to estimate link travel times, we formulate a probabilistic model for trip travel times. Let $X_k, k = 1, \ldots, M$ denote the random variables capturing travel times for the k^{th} link and $x_k = \mathbb{E}[X_k]$. Note that this model is separately built for weekday/weekend and for each hour of those days. Also, let T_p be the random variable denoting travel time for path p and $\bar{T}_p = \mathbb{E}_X[T_p]$. First we formulate the *general model*: $\sum_{k \in p} X_k = T_p + \eta$ where, η is a Gaussian noise with mean ϵ close to 0 and variance σ^2. Then taking expectation on both side it becomes

$$A\boldsymbol{x} = \bar{\boldsymbol{T}} + \epsilon \mathbf{1} \tag{1.1}$$

where $\boldsymbol{x} = [x_1, \ldots, x_M]^T$, $\bar{\boldsymbol{T}} = \left[\bar{T}_1, \ldots, \bar{T}_N\right]^T$, ϵ is the mean of the Gaussian noise, and $\mathbf{1}$ is the vector of all ones. $A_{N \times M}$ is a coefficient matrix, where $A_{ki} = 1$ if i^{th} link is a part of

k^{th} path, $A_{ki} = 0$ otherwise. We use maximum likelihood paradigm to estimate the mean link travel times x_k, with an additional constraint that $x_k \geq 0, \forall k = 1, \ldots, M$ as travel time can't be negative. The estimation problem then can be formulated as:

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmin}} \parallel A\hat{\boldsymbol{x}} - \bar{\boldsymbol{T}} \parallel_2 \quad \text{s.t} \quad \boldsymbol{x} \geq 0 \tag{1.2}$$

which is a least-squares estimation with constraints (LSEC). Note that this non-negative least squares problem [51] is a convex optimization problem, and can be solved using many efficient algorithms.

Variance in Link Travel Time Estimation

The *general model* described earlier can be written in a compact form as: $A\mathbf{X} = \mathbf{T} + \eta\mathbf{1}$ where $\mathbf{T} = [T_1, \ldots, T_N]^T$. Taking co-variance on both sides of the compact form, and assuming that the noise ϵ is independent of $\mathbf{T}$ and independent of each other, we get:

$$A\Sigma_X A^T = \Sigma_T + \sigma^2\mathbf{I} \tag{1.3}$$

where, Σ_X and Σ_T are covariances of $\mathbf{X}$ and $\mathbf{T}$ respectively and σ^2 is the variance of the Gaussian noise. Since both the matrices are very large to compute for our application, we assume that traffic in different links are independent of each other – all components of $\mathbf{X}$ and $\mathbf{T}$ are uncorrelated, resulting all the off-diagonal elements reduced to zero and the equation 1.3 becomes $A\boldsymbol{s} = \boldsymbol{S} + \sigma^2\mathbf{1}$ where $\boldsymbol{s}$ and $\boldsymbol{S}$ are the diagonals of Σ_X and Σ_T respectively. Here, we also assume that only variance of path travel times are used for estimation of $\boldsymbol{s}$. Since we are interested in finding s for the minimum variance in the noise, i.e. minimum σ^2, we get the following non-negative least-squared problem:

$$\boldsymbol{s}^* = \underset{\boldsymbol{s}}{\operatorname{argmin}} \parallel A\hat{\boldsymbol{s}} - \boldsymbol{S} \parallel_2 \quad \text{s.t} \quad \boldsymbol{s} \geq 0 \tag{1.4}$$

confirming the positivity constraint of the variance. We call this method *Independent Link Variance* (ILV) estimation. Also equations 4.4 and 4.8 are our **LSEC** methods i.e., least square estimation with constraints.

Route Recommendation

We further develop three route recommendation schemes based on (1) shortest distance (*RecDist*), (2) minimum expected travel duration (RecDur), and (3) minimum variance or most certain (reliable) route (*RecCert*); and compare them with Google Maps, Waze and most frequent path (MFP) [29]. The comparison shows that although distances and travel times for all the techniques are close, the actual paths proposed are significantly different. We see that if user chooses most certain path (using *RecCert*) then on an average, she needs to travel 0.09 mile more than the path recommended by *RecDist*, and she needs 0.32 minutes longer than the path suggested by *RecDur*. It also indicates that the average speed on the path recommended by *RecCert* is only 0.16 mile/hour shorter than that of the average speed on the path suggested by *RecDur*. Hence, it concludes that there is not much penalty (on an average) in choosing a more certain path.

1.3.2 Estimation of covariance in link travel time

Dynamic traffic demands result in change in travel time in different road links - causing heavy penalty on the transportation system. Often such changes lead to spread in congestion to adjacent as well as remote zones to the epicentre. Recent works assume that the travel time of a link is spatially correlated only with it's neighbouring links within a local area, which is not true in many practical scenario. Therefore, in this work a new method is proposed which doesn't put such proximity constraint and explore the method for estimating the covariance in link travel time using the path travel time from crowdsourced taxi trip data. Experiments show that our Dependent Link Variance (**DLV**) model is better by at least 12% than the baseline model. Our DLV model also predicts the impactful road links within an average error radius of ~ 0.14 mile and discovers correlated links which are more than 1.5 miles apart. Under the influence of city events, our model is able to predict the most affected paths within an average accuracy of 76.5%.

Dependent Link Variance Model

Our DLV method is extended from our previous work where we assume that traffic in different links are independent of each other. But in this work we build DLV method where we relax this assumption and take a more generic approach – traffic in different links are correlated. Therefore, from equation 1.3 in our previous work, we try to estimate Σ_X i.e., link covariance matrix directly without any simplifying assumption. So in this work, our final objective becomes, given a Σ_T for the path covariance matrix, estimate the link covariance Σ_X using the least square formulation:

$$\begin{aligned} \underset{\Sigma_X}{\text{minimize}} \quad & \|A(\Sigma_X)A^T - \Sigma_T\|_F^2 \\ \text{subject to} \quad & \Sigma_X \succeq 0 \end{aligned} \tag{1.5}$$

This problem is an instance of convex semidefinite program [7], where the constraint ensures semidefiniteness of the link covariance matrix Σ_X. Unfortunately, storing Σ_T, which is $N \times N$ dense matrix, requires exorbitant storage in the current scenario where $N \sim 10M$. Hence, solving the above problem directly is not possible, even with very high memory machines.

If the matrix A^TA is invertible, we can also write equation 1.3 as:

$$\Sigma_X = (A^TA)^{-1}A^T\Sigma_TA(A^TA)^{-1} + \sigma^2(A^TA)^{-1} \tag{1.6}$$

Here note that the matrix A^TA stores the counts of common paths between links, i.e. $(A^TA)_{i,j}$ = number of common paths between links i and j. Hence, it is expected that A^TA is a sparse matrix. We can assume that variance in noise is negligible ($\sigma \to 0$). Therefore, the final equation becomes

$$\Sigma_X = (A^TA)^{-1}A^T\Sigma_TA(A^TA)^{-1} \tag{1.7}$$

We call this our Dependent Link Variance (DLV) model. However, the above covariance function is not expected to be sparse. This has the problem of not finding important correlations which can be found using sparse inverse covariance estimation problem [17].

The sparse inverse covariance matrix W, can be estimated as:

$$\min_{W} \log |W| - Tr(W\Sigma_x) - \lambda \|W\|_1 \tag{1.8}$$

1.3.3 Prediction of taxi pickup hotspots during road closure incidents

In large cities taxi services normally maintain hotspots; these are either city-specified taxi stands or certain spots which have evolved over time. The knowledge of hotspots helps daily/experienced commuters to quickly and easily avail taxi service. Such hotspots may get severely disturbed during a road blockage (e.g. closure due to maintenance) around that area whereby taxis dynamically build new hotspots. These new hotspots may be far away from older hotspots, hence causing inconvenience to the commuters as well as resulting in loss of business and time of taxi drivers. A service which can predict the new hotspots and accordingly suggest its users to the most convenient new hotspot would be of tremendous value to both experienced and new commuters.

In order for the scheme to be successful certain conditions need to be fulfilled. First, one needs an automatic technique to identify such events, road activities causing the change in hotspots. Luckily nowadays information of such events widely get publicized in online social network. Therefore, we need to develop a scheme to identify them from the million other non-relevant information and accurately pinpoint the events, correct location. Second, in order to predict (change in) hotspots, one needs historical information of hotspots, not only the location but also their strength – how many pickup has been done from these hotspots. Fortunately this pickup information is available from various transportation datasets from different cities across the world like, Porto, Boston, Chicago, New York, etc., where these datasets are made public either by the city authority or by taxi hiring company (e.g. Uber[15]) from which hotspot learning is becoming possible.

In this work, we concentrate our study on New York City. This is because both disruption information as well as taxi hotspot information are available for this region. Moreover, on studying, it is found that there is significant 'movement of hotspots' during any disturbance. The New York City Department of Transportation (twitter handle: NYC_DOT)

[15]https://movement.uber.com/cities

regularly post tweets about various events in New York, out of which road maintenance, bridge closure is also announced. Next the New York City Taxi and Limousine Commission routinely releases information regarding trips of yellow taxi, their pick-up and drop-off points from which one can calculate the hotspots in the city.

Our observation shows $\sim 40.47\%$ repetition in geographic locations, where various road closure incidents take place, between year 2015 and 2016 in New York City. This clearly indicates that same location has been affected over the years and motivates us for hotspot prediction around the event location. To build the knowledge base, first we construct a classifier with an average accuracy of $\sim 94.55\%$ for identifying the relevant tweets having different road closure information. Next we extract the *date*, *time* and *location* information from those tweets and build our knowledge base. Second, leveraging the inferred knowledge, prediction of taxi pickup hotspot is done, near the activity location, with an average accuracy of $\sim 86.04\%$, where the predicted locations are within an average radius of only 0.011 mile from the original hotspots.

1.4 Organization of the Thesis

- **Chapter 1** introduces the basic motivation behind the work and presents an overview. The organization of the thesis is also elaborated here.
- **Chapter 2** presents a detailed literature survey of the related works (primarily from works in transportation network literature) on (a). different methods for link travel time prediction, (b). correlation estimation in link travel time, (c). change of traffic patterns during sudden road closure.
- **Chapter 3** presents methods to estimate link travel time and understand the reliability of a route in transportation network. It explains whether a shortest length path will always be the most reliable path or not. It also advises people regarding goodness of a path in terms of the path distance, duration and path certainty.
- **Chapter 4** depicts method for estimating the correlation in link travel time of city traffic – which is a key step in understanding how different road links impact each other. It also helps to understand the long range link correlations in travel time.

- **Chapter 5** describes methods for learning the geographical change in popular taxi pickup hotspots during various link disruption activity in the city. Online social network data along with transportation data is used to understand and predict such changes.

- **Chapter 6** finally concludes the thesis by summarizing all the works along with some possible scopes and extensions that can be drawn from the thesis.

Chapter 2

Related Works

The main areas of study in this thesis are: (1) how travel durations are distributed over road links, (2) the most reliable paths through a transportation network through traffic congestion, and (3) how traffic behaves during activities which cause road closure. In this chapter, we present a comprehensive survey of previously published papers related to the above areas in transportation networks. The organization of the survey is as follows: first (section 2.1), we present a brief overview of articles which have tried to estimate travel time in the transportation network under different settings; next, in section 2.3 we discuss about the papers which aim to figure out various correlations in the transportation network; finally, in section 2.4 we discuss about how heterogeneous data sources, such as online social networks and transportation networks, can help to understand traffic patterns, and present more meaningful insights related to traffic dynamics.

2.1 Travel Time estimation

The two main factors determining systems for estimation of link travel time are: data sources and the underlying model. One class of data sources includes loop detectors and automatic vehicle identification methods [3, 26, 35]. Here a loop detector is an electromagnetic inductive loop installed on the road to detect vehicle movement. When the motor

vehicle drives over the detector wire loop, the detector senses the metal by its electromagnetic wave. This can be used to determine the count as well as instantaneous velocity of the vehicles crossing it.

Li et al. [26] uses a linear model to predict travel times from flow. They assumed that travel time will be directly related to the traffic flow of the road network and consider a liner model using this flow. Asghari et al. [3] discusses the intrinsic variability in link travel times and predicts the most reliable route. They use the idea of probabilistic link travel time (*pltt*). They define it as follows: In a road network for every link l, the *pltt* is defined as the probability of taking x seconds for a vehicle to travel link l starting at time t. Pan et al. [35] does a speed prediction of vehicle using the sensor (loop detector) data. Where given a set of observed speed reading, they try to predict the speed for a short and long rage prediction horizon. They have combined the historical average model (HAL) with the Auto Regressive Integrated Moving Average (ARIMA) and proposed H-ARIMA using historical traffic pattern and current traffic speed.

Another set of data sources are low-frequency GPS probe vehicle data [41, 58, 64]. Here, GPS (global positioning system) is installed in the vehicle and it sends its current coordinate and timestamp to the server at a regular interval of time. Thus it creates a footprint for the vehicle with timestamp. Therefore, installing GPS in multiple vehicles, different observations of travel time for different road segments can be obtained and can be analysed for travel time prediction task. Zheng et al. [64], uses the Artificial Neural Network (ANN) model to estimate travel time. Here vehicle position, timestamp and speed are given as input to the model. However traffic flow information and signal timing are considered optionally as they were not available always. Rahmani et al. [41], predict travel times for routes, rather than links, thereby skipping the intricacies of path time models. They have used non-parametric kernel based approach for predicting the travel time for routes. Next Wang et al. [58] models the travel time of the drivers in different time points with three dimensional tensor. They fill in the missing values of the tensor from different contexts (like geospatial, temporal, historical) learned from trajectories and the map data. Ma et al. [30] estimates the probability distribution of trip travel times for an arbitrary OD pair at an arbitrary time given a set of probability distributions of link travel times. They use generalized Markov Chain approach, incorporating the conditional dependence of link travel times in the model formulation.

The main drawbacks of above mentioned data sources are that they are expensive to acquire since we need logistics and infrastructures for installing the loop detectors, GPS etc. Recently, many taxi hiring services like Uber, Ola, Lyft, etc., have come up, which track locations of their taxis in real time, through GPS, for scheduling customer pickups. Also different city authorities like Porto, Beijing, Boston, New York City, etc., have made their taxi data public for analysis with some anonymization for privacy preserving of their customer. This third kind of data source is cheap and easily available. Recently authors have started using these data sources for travel time estimation. [62, 63] try to estimate distributions of link travel times from this kind of crowdsourced taxi trip data. They use the fare field logged in the data to infer the probability of path taken by a given trip. Thus they form a *path probability estimation* (PPE) approach for estimating the link travel time. But these methods are computationally expensive and unlikely to be scalable to citywide data. They also use EM algorithm, which only gives a local optimum and hence is prone to differences in initial points.

2.2 Route Recommendation

One important problem while planning a travel is to find the best possible route. However it's not always intuitive that the shortest distance path will also take the minimum time to travel. Because more people tray to avail this path due to it's short distance, causing high congestion and greater travel time. Therefore, different proposals have been made for suggesting better routes. Google Maps is arguably one of the most used route recommendation system. They provide several alternate paths in increasing order of travel times. Google uses crowdsourced location data from the online users who are using their Google Maps app or web service. Thus they get real time movement of the app users. This helps them to estimates for link travel times [1]. However several people has raised concerns regarding privacy preservation. On the contrary Waze[1] takes different approach. Here users actively report to the traffic community regarding road accidents, road blocks, poor weather condition etc. Waze then collects this report, analyses and makes suggestions. It also recommends the fastest route, with crowd sourced annotations of links on the route. However,

[1] http://www.waze.com

these systems are based on individuals participation has the requires good infrastructure, e.g. presence of smartphone, good internet connectivity, higher social responsibility of the participants, etc, which is may not be available in many cities in developing countries.

Apart from these two popular systems there are other models as well. Chen et al. [13] returns the route with maximum probability. Where they assign transfer probabilities to intermediate locations which are significant and then take their product. Finally they return the route with highest probability. Wei et al. [59] provides a map-free heuristic for constructing most popular routes from uncertain trajectories. Given a sequence of locations and time interval, their method is able to build top k-routes using the uncertain trajectories given as input which sequentially pass through the locations within the specified time interval. They do this by aggregating such uncertain trajectories in a mutual reinforcement way i.e *uncertain* + *uncertain* $\rightarrow$ *certain*. Luo et al. [29] defines a *more frequent than* relation between footmark sequences, which characterize the frequencies of paths, and propose efficient algorithms for finding time-period based most frequent paths (MFP).

2.3 Correlations in Link Travel Time

In transportation network traffic dynamics evolve in a complex pattern. The behaviour of traffic in a road segment is correlated to various factors. Sometime any event (e.g parade, carnival etc.) happening nearby disturbs the traffic pattern in the local region or sometime natural calamity, earthquake, tsunami have devastated the transportation system of the entire city. Also a sudden change in traffic in any road segment due to road accident may change the traffic dynamics of the nearby road links. So there are various correlations in link travel time which needs to be understood.

Therefore, authors have tried to include traffic periodicity from the neighbourhood for estimating link travel time. Wang et al. [56] uses *non-route* based approach for travel time estimation and traffic dynamics interpretation. The authors propose neighbour based method considering only the neighbouring trips and include traffic dynamics by making use of traffic periodicity based on recent traffic patterns. Even the type of vehicles which are running through the road can decide the congestion of the road network. If a road only

allows public transport vehicles then the congestion type will be different than if the road allows both public transport and goods carrying large trucks. Because the goods carrying truck will move in a lower speed compared to the public transport resulting slow traffic flow in the road. Therefore Quin et al. [40] tries to model the link traffic using the interplay of multiple vehicle class and their behaviour in the overall link congestion. The authors claim that, when two classes of vehicles are mixed, each vehicle class influences the overall congestion in the road link depending upon their engine power, vehicle automation levels, etc.

However none of the above methods attempt to infer correlations in link travel time directly from path travel time. Jenelius et al. [20] separates trip travel times into link travel times and intersection delays and models correlation between link travel times on different network links using spatial moving average (SMA) structure. However, they assume that the correlations are related to externally observable factors, rather trying to infer them from large scale travel time data [33, 34]. Nie et al. [33] assumes that the link traversal time is conditional on the state of the preceding link, that is, they assume the state of a link to have a Markovian property. They think that the state of present link is dependent on the state of the link traversed right before arriving at its starting node, and independent of the links traversed prior to that. So only the consecutive links are correlated. But this concept has been extended by [11], which has assumed that the topological distance (measured by number of links) between any two correlated links can be upto k, thus extending the idea of adjacent link dependency.

2.4 Understanding Traffic from Multiple Data Source

Analysing transportation network is not only limited to investigating the transportation data only. Because various other factors like weather condition, government notifications also influence traffic. However all these information are broadcast in different channels like electronic new medium, online social networking (OSN) site etc. Therefore recently authors have focussed on multiple source of information for understanding meaningful traffic insights. This is a very crucial task as it helps to analyse and infer the reason of traffic disturbance from social media channel and then use that information for further investigation of the transportation data for finding out a solution and thus create a better travel experience

for city dwellers.

2.4.1 Using Homogeneous Online Social Network data

Today with the enormous growth of social network and mobile communication, people are acting as human sensors and sharing their travel experience in real time. Event extraction from social media data [5,6,43] is relatively robust technology, where authors try to extract event information from OSN feed. There are also works where researchers are specifically interested in finding out fruitful traffic insights from the social media channel and then further predict about it [38, 52]. Pathak et al. [38] shows how data from social network like facebook can deliver useful urban traffic information using multi-class classification technique to categorize traffic incidents. Tejaswin et al. [52] uses the spatio-temporal tweet feed data for traffic incidents clustering and prediction using Random Forest model. Mo et al. [31] focusses on the social media channel of the traffic police for understanding the road congestion time, place and proper reasoning using the linguistic dynamic system based on multi-factor time varying universe.

2.4.2 Computing with Heterogeneous Urban data:

Apart from using the online social network data only, authors have tried to combine other sensor data as well. Wang et al. [57] extends the traditional sensing paradigm by coupling sensors like GPS, mobile phone, humidity sensor with facebook and twitter, so as to facilitate real world event information fusion and analysis. Lécué et al. [24] shows how the severity of a road can be predicted combining heterogeneous, exogenous and raw data streams. Daly et al. [15] uses the dynamically occurring incidents like traffic accidents, localized weather conditions, unplanned obstructions from social media to provide user real time feedback to highlight the cause of the traffic disruption. Rashidi et al. [42] argues about the transportation aspect of heterogeneous mixture of social media data to infer various travel attributes. Pan et al. [36] first explores the traffic anomalies in the transportation data and then infer the reasoning of the anomalous behaviour from the social media content for further investigation. Finally Wu et al. [60] tries to combine ubiquitous data

sources, like point-of-interest (POI) from FourSquare, weather data, geo-tagged tweets and collision records for explaining daily traffic trend at some pre-decided locations by finding traffic correlation with these external data sources using machine learning technique.

Chapter 3

Link Travel Time Estimation and Route Recommendation

3.1 Introduction

Estimation of link travel time [44] is a core problem in transportation engineering, with applications ranging from detection of traffic behaviour and anomalies at different places and times [35], assessing and improving overall efficiency of transportation systems [44], to the widespread application of route recommendation for individual users [3], etc. Google Maps[1] provides excellent route recommendation, but relies on users being *online* for collection of traffic data; this is not sustainable in many developing countries, since connectivity is expensive, and not as abundant as in developed countries. Therefore, a more ideal scenario for any recommendation algorithm would be to rely on data collected/released by the vehicles (and not individuals) plying in the city.

In recent times, several cities around the world have started online taxi hire companies, such as Uber[2], Ola[3], etc., which collect route information. Besides, in many cities, the city administration makes it mandatory for taxi hiring services to log their trip informa-

[1] http://maps.google.com
[2] https://www.uber.com/
[3] https://www.olacabs.com/

tion. These taxi service organizations are increasingly making their taxi trip data public (e.g., Uber data[4]). However, understandably due to various constraints such as privacy concerns of the driver and the passenger, legal issues, business interests, etc., hardly any dataset contains complete trajectory information. For example, sometimes trip information is anonymized, or only pickup information is provided, or no distance information is shared, or exact pickup-dropoff information is masked. This trend is more or less common across taxi trip datasets released from different cities across the world like Porto, Boston, Chicago, New York, etc.

An important task, therefore, is to devise methods to meaningfully mine and utilize these increasingly available but incomplete trip datasets. In this work, we specifically take up this challenge and develop a methodology to tackle the situation where entire trajectory information is obfuscated. New York City Taxi and Limousine Commission[5] has freely shared vast amounts of data for about all taxi trips made in the city recently. The data however is incomplete and does not contain routes for trips, but only starting and ending locations and timestamps along with total distance travelled, fare, etc., for each taxi trip. We leverage on the huge amount of data provided to derive link level information of a large portion of links, and then use that for travel time estimation between any two given points. We believe the basic idea of *using vastness of data to reconstruct data that is not provided* is a different outlook as none of the previous travel time estimation algorithms work for such incomplete data [3, 26, 41, 64]. Note that our approach is different from missing data imputation approaches in statistics, which focus on substituting parts of data points which are absent.

In order to solve the problem on this dataset, following steps are undertaken: (a) Reconstruction of the route information from the (source, destination, distance) data, which is performed by enhancing the data with OSM[6] information; (b) From the route information, estimate the travel time for each individual link[7]; (c) For any arbitrary route, we predict the travel time by stitching back the time taken to travel its constituent links; (d) As an aside, we also find the variance in link travel time, which helps us to estimate the degree

[4]https://movement.uber.com/cities

[5]http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

[6]http://www.openstreetmap.org

[7]A link is a single segment of a road

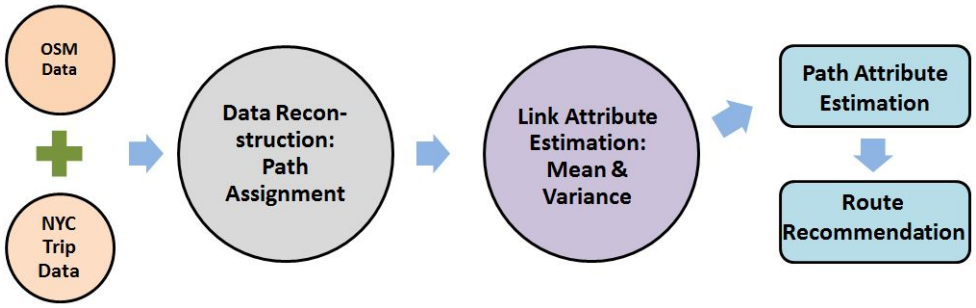


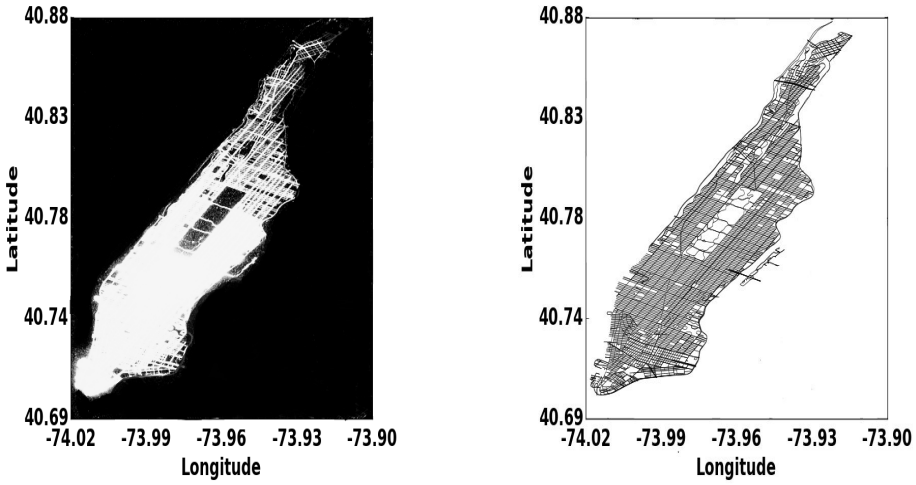
Figure 3.1: System Overview: Data reconstruction with the help of *NYC taxi trip* data and *OSM* data followed by link attribute estimation and application.

of uncertainty attached to a journey; (e) The estimated link attributes (mean, variance) are further used to build a recommendation system, which can either optimize time, distance, or certainty. Figure 3.1 shows an overview of the system developed.

Our framework maps 65.6 million taxi trips in Manhattan in 2014 within 0.1 mile error threshold to unambiguous (unique) paths (section 3.2). The link mean travel time, as well as its variance, are accordingly estimated using our **LSEC** method (section 3.3), as the path travel time is spatially distributed over the constituent links of the paths. Results from experiments (section 5.4) with link travel time prediction algorithms, show that mean absolute percentage error (MAPE) for predicted link travel times is better by at least 20% than existing methods. Also, the root mean square error for path variance estimation is within at most 1.5 minutes on an average, i.e., for an observed path variance of 5 minutes, our estimated variance lies within $3.5 - 6.5$ minutes, confirming the effectiveness of the variance estimation method. We further develop three route recommendation schemes (section 3.6): shortest distance (*RecDist*), minimum expected travel time (*RecDur*), and minimum variance or highest certainty route (*RecCert*); and compare them with Google Maps, Waze and most frequent path (MFP) [29]. The comparison shows that although distances and travel times for all the techniques are close, the actual paths proposed are significantly different.

3.2 Dataset Construction: Assign paths to trips

In this section, we describe the construction of the *route dataset* that is, assigning a unique path from the road network to a taxi trip from the NYC dataset. Note that all the given trips may not be assigned to a unique path as there may be multiple paths with roughly same distance between two points. However, the presence of huge amount of data ensures even if a large number of pairs cannot be uniquely mapped, we can still glean out a sizeable amount of data (route information) with high confidence. These route-annotated data can then be used to estimate route-travel time described in next section. The detail steps for route-annotation is described next.



(a) (b)

Figure 3.2: (a) Original pickup, dropoff distribution across Manhattan (b) Constructed full road network for Manhattan

3.2.1 Dataset

NYC taxi dataset: We obtained this data from the New York City Taxi and Limousine Commission. For this work we have focused only on Manhattan City and consider only those trips which start and end inside the Manhattan City itself. After initial data preprocessing, cleaning and filtering, it has **86.1** million valid trips made in Manhattan for the year

Field	Meaning
pickup date time	trip start time
dropoff date time	trip end time
trip time	total trip duration (in sec)
trip distance	total distance covered (in mile)
pickup location	GPS coordinates
dropoff location	GPS coordinates

Table 3.1: Features from taxi trip data used in this work

2014. However the trip data doesn't provide intermediate route information for any trip. It provides only the pickup and dropoff coordinates, timestamp along with total trip distance and duration. We have used only the following features: *pickup and dropoff timestamps, coordinates, trip distance and duration* only.

OSM dataset: While road networks are available for many cities, from the city corporation or state transportation authorities, very often these networks are incomplete or have missing properties [53]. Following [53], we use data from OpenStreetMap (OSM), which is an open source collaborative project dedicated to create free and editable map of the world. We download the OSM data for Manhattan city (longitude range: $(-74.02, -73.90)$ and latitude range: $(40.69, 40.88)$) and have created the corresponding road network. Figure 5.1 shows the road network for Manhattan city, which has **14886** nodes and **23884** links. Next we describe construction of road network from OSM data.

3.2.2 Road Network Construction using OSM

In this section, we describe our road network construction algorithm from OSM data [53], having various structural *elements* e.g. *node*, *way*, etc. For our map generation we need two types of *elements* namely: *node* and *way*. A *node* is a tuple of $(longitude, latitude)$ and *way* is an *ordered set of referred nodes* that represents linear or curved structures in the map. In OSM data, a *way* element has *tag: "highway"* with different *tag_value* namely "primary","secondary","tertiary", etc., denoting the road type for that *way*. Similarly if a *way* is one directional then it has the *tag: "oneway"* and *tag_value:yes*. Following the tech-

nique used by Toole et al. [53], we use the following steps to construct the road network. First we filter out all the *way* element whose "highway" tag has the values: "primary", "secondary", "tertiary", "unclassified", "road", "living-street" and "residential" only and create the $filteredWaySet$. Second we take union of all *way* element in the $filteredWaySet$ and construct the $nodeSet$. Third we take the union of all the directed links (adjacent ordered node-pairs in the *way* element) in all the *way* elements from $filteredWaySet$ and construct the $directedLinkSet$.

Finally using $nodeSet$ and $directedLinkSet$, we create the final road network. Figure 5.1-(b) shows the constructed road network for Manhattan City using the above algorithm and the data from OSM. This road network has 14886 nodes and 23884 links. We note that the number of nodes can be reduced by just considering junctions of road segments. However, we keep the superficial (extra) nodes as these help in: (a) capturing the curvature of curved road segments and (b) increasing the accuracy of mapping trip data to the road network in next section. We also plot in figure 5.1-(a), the pickup and dropoff locations on the map to understand the spatial density of trips in Manhattan city. Here *white* region denotes that too many trips occurred in those area over the year.

3.2.3 Mapping Taxi Trip data to Road Network

We use *proximity based location mapping*: to map all the original location to the closest node (according to Euclidean distance) in the road network. In the subsequent analysis, the original locations in the trips data, are replaced with the closest node in the road network. A total of 7028049 locations (resulting from taxi (pickup,dropoff) pairs) were mapped to **14886** nodes in the road network. Figure 3.3 plots the euclidean distance between the original (from NYC taxi data) and the mapped locations (from OSM data) (x-axis) against the cumulative fraction of points (y-axis). This distance measures the *error* in the mapping process. We can see that nearly **99%** of 7028049 locations are mapped with an error of only **0.08 mile** or less. The superficial (extra) nodes in the road network which capture the curvature of road segments contributed significantly in producing such highly accurate results.

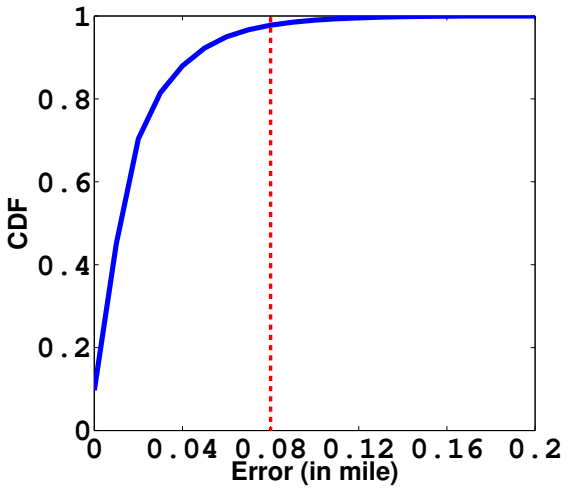


Figure 3.3: *CDF* shows 99% taxi OD pairs are mapped to the closest node within an error threshold of 0.08 mile

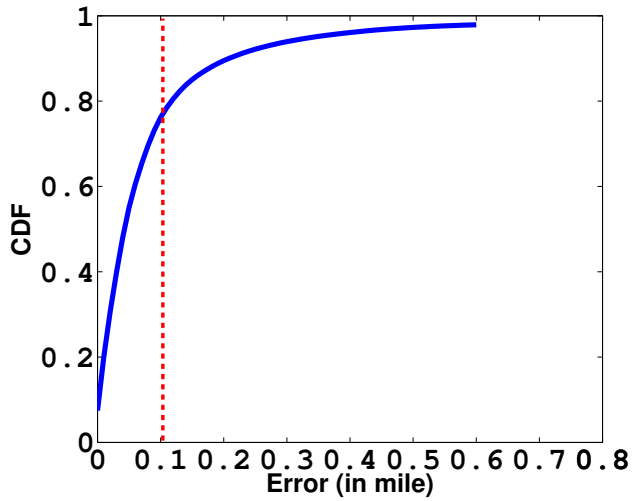


Figure 3.4: *CDF* shows 76.1% trips are attributed to unique paths within 0.1 mile error threshold

3.2.4 Distance based Path Attribution

The biggest challenge here is that the path taken through the road network between origin-destination (OD) pairs is not provided. Recent works [62, 63] use probabilistic models to estimate the probability of alternative paths being taken. However as mentioned above, these approaches suffer from many drawbacks, including arriving at negative link travel

time estimates. In this work, we take an orthogonal approach: *distance based path attribution*. Our main idea is to filter out trips for which there are multiple paths with very low error in the total distance travelled. This leaves us with a dataset of trips which can be attributed to unique paths with low error. This strategy can be especially effective, given the large number of trips in the dataset. Algorithm 1 describes the steps.

Algorithm 1: Distance to path matching

Require: Road Network $RdNet$, Trips data TD

1: $ODP \leftarrow$ All OD Pairs (TD)
2: **for all** $m \in ODP$ **do**
3: $R \leftarrow k$-shortest paths($m, RdNet$)
4: $fltR \leftarrow$ non-ambiguous paths(R)
5: **for all** $t \in$ trips_ODP(m, TD) **do**
6: {Trips between OD pair m}
7: $Path(t) \leftarrow CDM(t, fltR)$
8: $Error(t) \leftarrow |Dist(Path(t)) - TripDist(t)|$
9: **end for**
10: **end for**

return $Path, Error$

We generate k-shortest paths between each OD pair using Yen's k-shortest path algorithm [61], taking $k = 50$. For each path, we calculate the distance as sum of distances of the links from road network. The distances of all paths are then clustered into *distance buckets*. If multiple paths fall inside same distance bucket for an OD pair then we call them *ambiguous*. Step 4 filters out the non-ambiguous paths for a given OD pair (m) and stores it in $fltR$. In step 7, the function $CDM(t, fltR)$ returns the path which has a distance closest to the trip distance for trip t. Here closeness is measured using $Error$ as absolute value of difference in distances. In step 8, $Dist(p)$ gives the distance for path p, and $TripDist(t)$ gives the trip distance for trip t. The outputs of the algorithm are arrays $Path$ and $Error$. $Path(t)$ gives the distance-wise-closest non-ambiguous path for trip t and $Error(t)$ gives the corresponding error.

We applied the above algorithm for attributing paths to the taxi-trip data filtered to Manhattan area. Note that the algorithm attributes paths to all trips, unless the corresponding

OD pair has no non-ambiguous path. Figure 3.4 plots the *cdf* for errors (in mile) computed in the attributed paths. We can see that a large fraction of trips are attributed paths within an error of 0.1 mile, after which the rise in fraction of trips tapers off and the error in attribution increases. We choose 0.1 mile as the error threshold. **65.6 million** trips, out of total of 86.1 million trips in the Manhattan area, lie within an error threshold of 0.1 miles (retention of 76.1%). We consider this filtered and path attributed set of trip as our *route annotated data*. We use this data for link travel time estimation (sections 3.3) and route recommendation (section 3.6).

3.2.5 Distance based Path Assignment

After mapping to the road network, each trip in the NYC taxidata contains a pair of origin and destination nodes, called OD pair. The main problem is to map an OD pair to an unambiguous path in the road network, upto a given tolerance. The algorithm generates k-shortest paths between each OD pair using Yen's k-shortest path algorithm [61], and clusters the paths into buckets based on distance on the road network using threshold 0.01 miles. Paths mapped to buckets having more than one path are called *ambiguous*. Trips in NYC taxi-data also come with logged distances, which are mapped to the closest non-ambiguous distance bucket. The assumption behind the algorithm is that the unambiguous paths are present within the k-shortest path. We use $k = 50$. The algorithm is described in section 3.2.4 of appendix. We applied the above algorithm for attributing paths to the NYC taxi-data filtered to Manhattan area. Figure 3.4 plots the CDF of errors, the absolute difference between the trip distance in the NYC taxi-data and the distance computed on the attributed paths. We observe that a 76.1% of trips are attributed to unique paths, with error less than 0.1 mile. We choose 0.1 mile as the error threshold resulting **65.6 million** trips, out of total of 86.1 million trips in the Manhattan area to lie within this error threshold (retention of 76.1%). We consider this filtered and path attributed set of trips as our *ground truth trip data* for link travel time estimation (sections 3.3) and further application (section 3.6).

ETH (mile)		0.1	0.2	0.3	0.4
(%) of Link	NW	**84.97**	85.71	86.27	86.37
Coverage	W	**87.48**	88.28	88.53	88.76

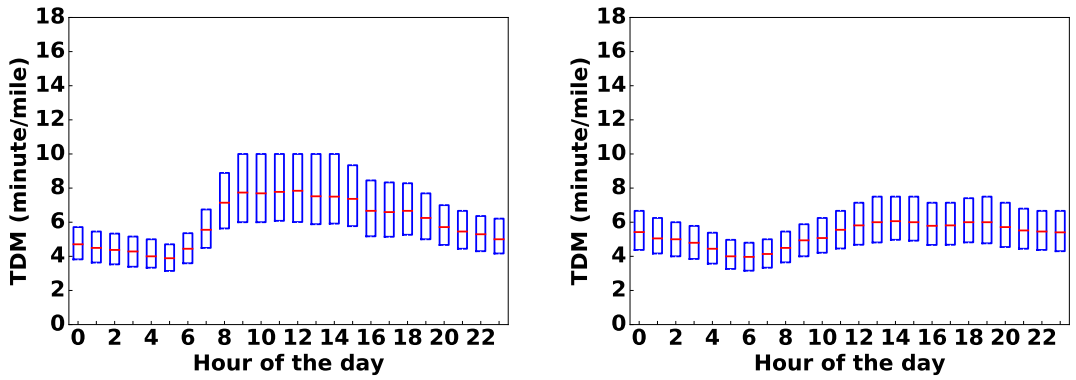
Table 3.2: Variation in link coverage (%) of road the network of Manhattan City with different error threshold (ETH) (in mile) for Weekends (NW) and Weekdays (W)

3.2.6 Characteristics of Trip Data

The trip dataset derived above is next used to extract link travel time related parameters. In order to derive link level information from path level information we need to consider the performance of a set of path travelled. However, for any stochastic method to obtain accurate values, the set of paths considered needs to follow similar distribution – at least, the average travel time per unit distance need to be roughly same for all. To ensure that here we report the link coverage and the hourly statistics of weekend and weekday traffic for appropriate bucket choice.

Link coverage: Once the paths have been attributed to trips, we measure the *link coverage*, which is defined as the fraction of links which are part of attributed paths to at least one of the trips. Table 3.2 reports the percentage (%) in link coverage for Manhattan areas, as a function of the error threshold (in mile) used for selecting trips. We see, using error threshold of 0.1 mile, **84.97%** and **87.48%** links have been covered for weekends and weekdays respectively.

Temporal Characteristics: We found that the variance within an hour remains minimum and it shoots up if we consider wider bucket. If we use wider bucket of size 2 to 5 hours, then the *within-bucket* variance increases from 12.5% to 58.38%. Hence while calculating the link travel time, we consider buckets of one hour duration. We find, TDM is low from midnight to 6:00 am in the morning, gradually increases from 6:00 am towards 10:00 am, remains stable from 10:00 am to 4:00 pm, and drops gradually there after. However the trend is slight different for weekends as shown in the figure3.5 where variance in TDM for late night is higher than weekdays. This indicates how variance in TDM is affected in



(a) (b)

Figure 3.5: Average daily trend of trip duration per mile (TDM) for year 2014 shows specific *stable-rise-stable-fall* trend and clear distinction between (a) Weekdays (b) Weekends.

24 hour cycle due to variation of traffic throughout the day. We therefore, consider four *one hour* window starting at 6:00 am, 10:00 am, 4:00 pm, 8:00 pm, for the subsequent experiments shown in sections 5.4 and 3.6.

Number of trips per path: Out of 9.6 million paths which are traversed between 5.3 million OD pairs, 4.5 million paths have a single trip. These paths are discarded while computing path variance (section 3.3.2), thus losing 47.57% of paths. The presence of huge data again ensures that even after discarding a large set of data, we are still left with a sizeable set to calculate variance besides mean link travel time (Table 3.5).

3.3 Methodology: Link Attribute Estimation

In this section, we propose our algorithms for estimating link travel times and variance in link travel time from start-to-end travel time data for many trips through the road network. The main challenge is that we do not directly know how the travel times are split across links of the path. This information is present in an indirect form as paths overlap on certain links. Hence, all the link travel times have to be inferred jointly. Also, there are two types of randomness: first is the noise in measurement of path travel times, and second is the intrinsic variability in the link travel times for each link. In order to minimize the intrinsic

variability, we bin the entire dataset into one hour buckets and calculate the link travel and variation time for each link. This is in line with the observation reported in [62].

Let $G = (V, E)$ denote the road network, where V is the set of all *nodes* which are the origin and destination points of various trips, $E \subseteq V \times V$ is the set of all road *links* in the network. We assume a directed graph with $e = (v_1, v_2)$ implying that traffic can flow from v_1 to v_2. A path p is an ordered list of nodes (V_p) such that $(v_{i-1}, v_i) \in E, \forall i$. Equivalently, it can also be thought of as a list of links (E_p) such that if $e_{i-1} = (u_{i-1}, v_{i-1})$ and $e_i = (u_i, v_i)$, then $u_i = v_{i-1}, \forall i = 1, \ldots, M$, where M is the total number of links in the road network. Our processed NYC trips dataset $\mathcal{D}$ obtained in section 3.2, can be described as a collection of paths p_i along with trip travel time T_i, $\forall i = 1, \ldots, N$, where N is the total number of of trips. Note that a given path p_i may have multiple trips. Hence, we can compute the mean travel time $\bar{T}_p$ for a path p as $\bar{T}_p = \frac{1}{N_p} \sum_{i:p_i=p} T_i$, where N_p is the number of trips through path p. Similarly, we can compute the total variance S_p in travel time for path p as: $S_p = \frac{1}{N_p - 1} \sum_{i:p_i=p} (T_i - \bar{T}_p)^2$. Next, we describe our methods for probabilistic link travel time and variance estimation.

3.3.1 Link Travel Time Estimation

In order to estimate link travel times, we formulate a probabilistic model for trip travel times. Let $X_k, k = 1, \ldots, M$ denote the random variables capturing travel times for the k^{th} link. Note that this model is for weekday/weekend and an hour of the day. Also, let T_p be the random variable denoting travel time for path p. We formulate the model:

$$\sum_{k \in p} X_k = T_p + \eta \tag{3.1}$$

where, η is a Gaussian noise with mean ϵ close to 0 and variance σ^2. This is a simplified model, which ignores waiting time at the nodes. Taking expectation w.r.t. all random variables X_k and noting that $x_k = \mathbb{E}[X_k]$ and $\bar{T}_p = \mathbb{E}_X[T_p]$:

$$\sum_{k \in p} x_k = \bar{T}_p + \epsilon \tag{3.2}$$

We also assume the various trip travel times T_i are due to variation in link travel times X_k. Hence, the average of trip travel times $\bar{T}_p$ for a path p can be used as a plug-in estimate for $\mathbb{E}_X[T_p]$.

With this model, and the data for path travel times $\{\bar{T}_1, \ldots, \bar{T}_N\}$, we can write the above equations in a compact form as:

$$A\boldsymbol{x} = \bar{\boldsymbol{T}} + \epsilon\mathbf{1} \tag{3.3}$$

where $\boldsymbol{x} = [x_1, \ldots, x_M]^T$, $\bar{\boldsymbol{T}} = \left[\bar{T}_1, \ldots, \bar{T}_N\right]^T$, and $\mathbf{1}$ is the vector of all ones. $A_{M\times N}$ is a coefficient matrix, where $A_{ik} = 1$ if i^{th} link is a part of k^{th} path, $A_{ik} = 0$ otherwise. Since we are interested in estimating x_k such that $\epsilon \to 0$ in the Gaussian noise η, we use maximum likelihood paradigm to estimate the mean link travel times x_k. This leads to the least squares problem:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \|A\mathbf{x} - \bar{\mathbf{T}}\|_2 \tag{3.4}$$

We call this the least-square estimate (LSE) for link travel times.

A problem with this formulation is that the optimal mean link travel time estimates x_k^* can be negative. This is because $\sum_{k\in p} x_k$ can be positive, even though the individual x_k are not all positive. Hence, we impose the further constraint that $x_k \geq 0, \forall k = 1, \ldots, M$. The final estimation problem becomes:

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmin}} \parallel A\hat{\boldsymbol{x}} - \bar{\boldsymbol{T}} \parallel_2 \quad \text{s.t} \quad \boldsymbol{x} \geq 0 \tag{3.5}$$

We call this the least-squares estimation with constraints (**LSEC**). Note that this is an instance of non-negative least squares problem [51], which is a convex optimization problem, and can solved using many efficient algorithms. Next we estimate variances in link travel times.

3.3.2 Variance in Link Travel Time Estimation

The equation 3.1 can be written in a compact form as:

$$A\mathbf{X} = \mathbf{T} + \eta\mathbf{1} \tag{3.6}$$

where $\mathbf{T} = [T_1, \ldots, T_N]^T$. Taking co-variance on both sides, and assuming that the noise ϵ is independent of $\mathbf{T}$ and independent of each other, we get:

$$A\Sigma_X A^T = \Sigma_T + \sigma^2 \mathbf{I} \tag{3.7}$$

where, Σ_X and Σ_T are covariances of $\mathbf{X}$ and $\mathbf{T}$, respectively. Unfortunately, both matrices are very large for our application. Moreover estimating Σ_T is a challenge give the dataset.

First we make a simplifying assumption that all components of $\mathbf{X}$ and $\mathbf{T}$ are independent. Hence, $\Sigma_X = diag(\mathbf{s})$ and $\Sigma_T = diag(\mathbf{S})$, where $\boldsymbol{s} = [s_1, \ldots, s_M]^T$ and $\boldsymbol{S} = [S_1, \ldots, S_N]^T$ are variances of $\mathbf{X}$ and $\mathbf{T}$, respectively. The above equation can be written as (since $(A\Sigma_X A^T)_{i,i} = \sum_{k=1}^{M} s_k A_{i,k}^2$ and $A_{i,k} = A_{i,k}^2$) :

$$A\boldsymbol{s} = \boldsymbol{S} + \sigma^2 \mathbf{1} \tag{3.8}$$

Since we are interested in finding $\mathbf{s}$ for the minimum variance in the noise, i.e. minimum σ^2, we get the following non-negative least-squared problem:

$$\boldsymbol{s}^* = \underset{\boldsymbol{s}}{\operatorname{argmin}} \parallel A\hat{\boldsymbol{s}} - \boldsymbol{S} \parallel_2 \quad \text{s.t} \quad \boldsymbol{s} \geq 0 \tag{3.9}$$

Note that the estimation problem is same as the link travel time estimation, except that the target value is path variance instead of path travel time. We call this method *independent link variance* (ILV) estimation. After solving equations 4.4 and 4.8, we compute the predicted **path travel time** for path p as: $T_p = \sum_{x \in p} x_k^*$ and the **variance in path travel time** for path p as: $s_p = \sum_{k \in p} s_k^*$, due to independence.

3.4 Experimental Setup

We compare the proposed least squares estimation with constraints (**LSEC**) method for estimation of link travel times with two baseline algorithms: (1) the path probability estimation based algorithm (PPE) proposed in [62] and (2) the naive least squares estimation based algorithm without constraints (LSE).

Baseline methods: PPE [62], assumes a linear model for fare in terms of distance and time, and tries to fit the coefficients using least squares regression. They assume a multinomial logistic regression model for route probabilities in terms of distance and time, using the fitted coefficients. Using the path probability, they compute expected travel time, which is matched to actual travel time for a non-linear least squares fitting. Our second baseline is basically the naive least squares estimation based algorithm LSE without positivity constraints on estimated links as described in Section 3.3, equation (3.4). While PPE needs the trip duration, distance and fare information, LSE and LSEC need trip duration and distance along with attributed paths.

One week dataset: Since optimization suggested in [62], uses the Jacobian matrix which is $O(|E|^2)$, PPE cannot be run on the entire road network. Hence, following [62], we select a smaller region of Manhattan City with 1829 nodes and 2334 road links for baseline comparison. Next we construct a smaller taxi trip dataset for running their algorithm. While the authors have used single week's data for the year 2010 we have used it for year 2014. They used the trip data from 3/15/2010 to 3/21/2010 where as we use our path attributed taxi trip data from 3/15/2014 to 3/21/2014 whose pickup and dropoff fall into test region and got total 155601 trips for these seven days period.

Experimental procedure: We compared with baseline using one week dataset (section 3.5.1) and validated the performance of our method on the entire 2014 dataset separately (section 3.5.2). We randomly select (80%) training and (20%) test set data and evaluation was done on test set data. All the work has been performed on machine having 48 CPU (Intel(R) Xeon(R) CPU, $E5-2697$ v2 @ 2.70GHz), RAM 256 GB. Distance based path matching took roughly $\sim 10^{-3}$ seconds per trip. For the road network with 23884 link, our LSEC takes on an average ~ 0.60 seconds per link for link mean and variance in travel time estimation.

Evaluation metrics: Let $\bar{T_p}$ be the observed path mean time for all the trips made by the path p and N_p be the total number of paths. Also let $\hat{x_k}$ be the estimated link mean travel time for k^{th} link in path p. Hence, estimated mean path time can be calculated as:

$\hat{T}_p = \sum_{k \in p} \hat{x_k}$. Next for comparing $\bar{T}_p$ with $\hat{T}_p$, we define the following metrics:

$$RMSE = \sqrt{\frac{\sum_p (|\bar{T}_p - \hat{T}_p|)^2}{N_p}} \tag{3.10}$$

$$MAPE = \frac{1}{N_p} \sum\nolimits_{p \in P} \left| \frac{\bar{T}_p - \hat{T}_p}{\bar{T}_p} \right| * 100\% \tag{3.11}$$

Similarly as described above we define the error measurement metrics for path SD estimation. Let σ_p be the observed standard deviation for the p^{th} path and $V_p = \sigma_p^2$ be the corresponding variance. Also, let $\hat{V}_p$ be the estimated variance for path p, N_p be the total number of paths and S_k be the estimated variance for link k. Then $\hat{V}_p = \sum_{k \in p} S_k^2$. The equations 3.10, 3.11 are redefined with replacing $\bar{T}_p$ by σ_p and replacing $\hat{T}_p$ by $\sqrt{\hat{V}_p}$.

3.5 Results: Travel Time Estimation

In this section, we validate the link travel time and variance estimation techniques proposed in section 3.3 using metrics defined above. In section 3.5.1, we describe results comparing **LSEC** with baselines (PPE [62] and LSE) on the one week dataset as described above. We also report the fraction of links with negative mean travel time. Note that we can only compare expected route travel times since authors of PPE do not prescribe a method for calculating standard deviation. In section 3.5.2, we report performance of our methods on the full dataset for link travel time and link SD estimation.

3.5.1 Baseline Comparison

Table 3.3 shows root mean square error (RMSE) and mean absolute percentage error (MAPE) for link travel time estimated by **LSEC**, LSE and PPE for four different time of the day 6:00 am (6), 10:00 am (10), 4:00 pm (16), and 8:00 pm (20). We note that LSEC performs nearly similar to LSE while it performs much better than PPE, both in terms of RMSE and MAPE. We observe, RMSE is higher for rush hours 10 and 16 when the average trip duration per

Hour	PPE		LSE		LSEC	
	RMSE (minute)	**MAPE (%)**	**RMSE (minute)**	**MAPE (%)**	**RMSE (minute)**	**MAPE (%)**
6	1.56	25.68	1.13	20.14	**1.12**	**19.15**
10	3.09	29.25	2.53	22.21	**2.26**	**21.57**
16	2.71	30.16	2.29	23.34	**2.12**	**22.26**
20	1.76	26.28	1.33	20.43	**1.29**	**20.12**

Table 3.3: Comparison of method LSEC with baseline PPE and LSE. LSEC is showing good progress over PPE in terms of both RMSE and MAPE.

mile (TDM) is higher. While for hour 6 and 20 the error is comparatively low supporting the TDM trend as discussed earlier in section 3.2.6 (Figure 3.5). The approach proposed in PPE suffers from several problems. First, linear relationship between fare distance and time could not be verified for the full one year dataset (they verify it with trip data of seven days spanned over a very small part of Manhattan). Second, it assumes that probability of following a path is related to the fare, which is not verified for larger city wide dataset. Third, the approach is computationally very expensive as it maintains probabilities for top-k paths for all trips, and hence could not be applied to the entire dataset.

Hour		**6**	**10**	**16**	**20**
(%) of -ve	**PPE**	23.73	24.25	27.97	24.29
Link time	**LSE**	12.12	15.81	14.65	13.24

Table 3.4: Comparison of percentage (%) negative link between PPE and LSE. For each of the hours CDF of LSE is always lower than that of PPE

Effect of negative links: A major drawback of both the baselines is that they have no mechanism to stop estimating negative link travel times. Table 3.4 reports the percentage of links with estimated negative link travel times. We note that both LSE and PPE estimate a significant percentage of negative links, and cannot be used for route recommendations.

We also investigate the correlation between percentage of negative links and the accuracy of travel time prediction. In figure 3.6 we have plotted the CDF of negative links frequency for different hours of the day for LSE and PPE. We see that the frequency of

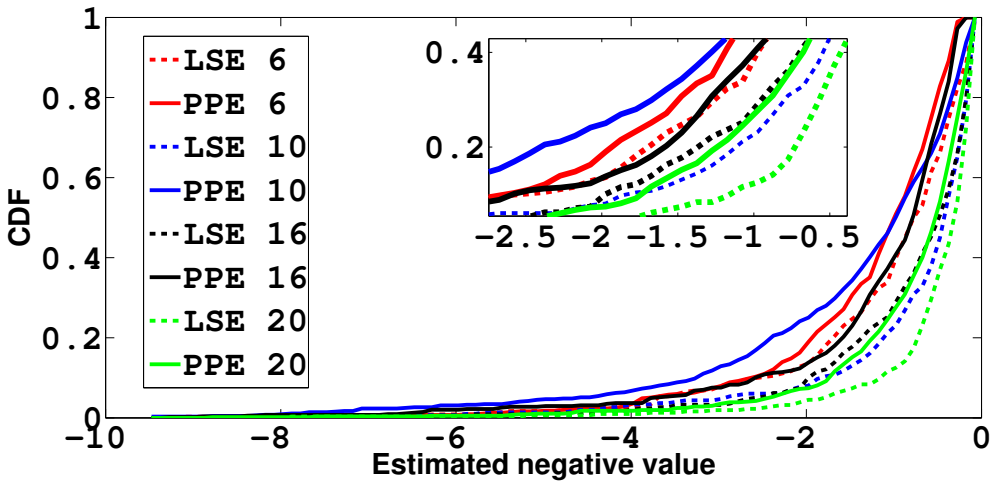


Figure 3.6: Comparison of frequency distribution of estimated negative links for PPE and LSE for different hours of the day

negative values estimated by PPE (given by solid lines) is more than those of LSE (given by dotted lines), which correlates with the inferior performance of PPE compared to LSE. Hence, we conclude that one of the reasons behind the inferior performance of PPE over LSE (and also LSE over LSEC) is the spurious computation of negative link travel times.

3.5.2 Performance of LSEC on Entire Dataset

Table 3.5 details the performance of **LSEC** which we discuss one by one. The last column in the table shows the total trip data available.

Validating average link travel time estimation: We see in table 3.5, for estimating path time by our **LSEC** method, $MAPE$ value is within 24% - intrinsic variability in the travel time data is partially responsible for the error. The undershoot and overshoot of estimation are roughly in similar proportion. Notice that although the MAPE is slightly worse in the weekend than the weekdays, the RMSE shows opposite behaviour because the car speed is higher in weekend (people reach their destination faster). Hence although the percentage error is high, the absolute mismatch with respect to time is low. The table also shows the number of estimated links over Manhattan city - the experiment covers. It is nearly 80% and 78% for weekdays and weekends respectively, clearly signifying the scale of our

Hour	Day Type	Path Mean Time Estimation				Path Time SD Estimation				TC
		MAPE (%)	RMSE (min)	PC (TC>0)	LC (%)	MAPE (%)	RMSE (min)	PC (TC>1)	LC (%)	
6	W	19.10	1.57	120640	77.06	37.19	0.62	65461	51.75	1434065
10	W	23.00	3.36	372090	79.57	32.17	1.50	199478	50.78	3166472
16	W	23.34	3.37	316246	80.96	31.72	1.43	164231	48.85	2503511
20	W	19.62	2.27	534513	84.61	35.44	0.99	269513	55.34	3639485
6	NW	21.23	1.47	57457	73.56	36.11	0.59	35823	35.12	184571
10	NW	20.10	2.14	221421	78.57	37.33	0.80	102284	47.39	1017649
16	NW	23.68	3.00	229880	80.06	33.60	1.08	102411	42.91	1054417
20	NW	22.51	2.60	228005	81.26	35.20	0.95	125697	43.48	1210975

Table 3.5: Measuring error in estimation of path mean travel time and path standard deviation (SD) estimation for *weekdays* (*W*) and *weekends* (*NW*). It also shows hourly details about Trip Count (TC), Path Count (PC) for (TC > 0 and > 1) during Weekdays (W) and Weekends (NW) and the corresponding Link Coverage (LC)

experiment.

Validating link variance estimation: Table 3.5 also shows the performance of **LSEC** for path SD estimation (via aggregating estimated link SD). Here $MAPE$ is within on an average 34% and 35% for weekdays (W) and weekends(NW) respectively. This indicates that the predictions of SD is not as good as link travel time. This can be attributed to the sparsity of data as in order to calculate variance of a link, a path needs to be traversed multiple times. We find that such paths are substantially less (column 9 in the table), the value becomes even smaller if we try to put a threshold of (say) minimum 10 trips to remove noise. Due to reduction in dataset the link coverage also reduces to around 50% and 40% in weekdays and weekend respectively.

3.6 Application: Route Recommendation

The huge amount of link level information (both mean travel time and its variance) estimated can be used to recommend routes by simply stitching the links having minimum

estimated value. The path suggested by minimum variance links denotes the path with less variability or uncertainty which motivates us to go for *certainty* based route recommendation. Commercial route recommendation systems, e.g. Google Maps or Waze, report two aspects of recommended routes: distance and travel time. Moreover, often there are shorter distance paths with heavy traffic resulting in more travel time or longer distance paths with less congestion resulting in lower travel time. Hence, there is no obvious choice. In this section, we propose a third parameter for recommending paths: **uncertainty** or intrinsic variance in path travel times and compare them with existing commercial system.

System	RecDist	RecDur	RecCert
Avg dist (mile)	**1.92** (0)	1.99 (0.07)	2.01 (**0.09**)
Avg dur (min)	12.15 (0.29)	**11.86** (0)	12.18 (**0.32**)
Avg SD	2.80 (0.23)	2.79 (0.22)	**2.57** (0)
Avg speed (mile/hr)	9.48 (0.58)	10.06 (0)	9.90 (**0.16**)

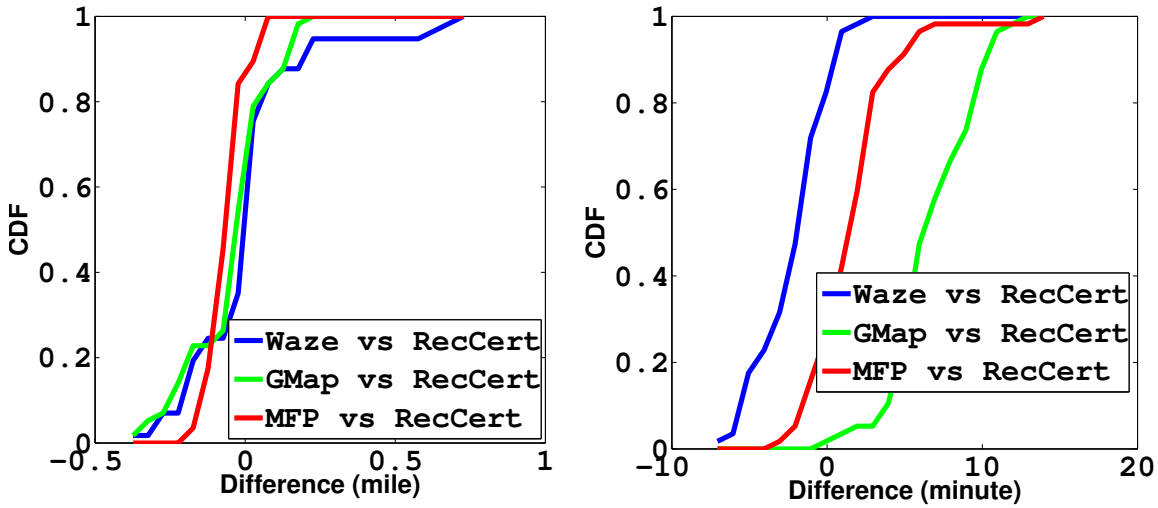
Table 3.6: Effect on speed due to different recommendation system. The value in the bracket shows how much it deviates from the optimal value

3.6.1 Recommendation Methodology

Given an origin – destination (OD) pair (u, v), *RecDist* recommends the lowest distance path between u and v, *RecDur* recommends the lowest expected travel time path and *RecCert* recommends lowest estimated variance path between the same u and v. We note that for any path p the path travel time T_p can be calculated as sum over the link travel times: $T_p = \sum_{e_i \in p} T_{e_i}$. This is also the same for the path distance and path variance (due to independence assumption (ILV) in section 3.3). The link distances are calculated from coordinates in OSM data, whereas the link mean travel times and variances are calculated using LSEC and ILV methods developed in section 3.3, using the full training taxi trip dataset. Since the individual links are always positive, hence we use Dijkstra's shortest path algorithm [14] to choose shortest duration, distance, or variance path accordingly.

Results: For each OD pair we recommend 3 optimal paths using the three recommendation schemes described above and note down the estimated distance, duration and SD

for the recommended path and calculate the average speed [62]. In table 3.6 the numbers in bracket report improvement over recommendation in the category. We see that if user chooses most certain path (*RecCert*) then on an average, she needs 0.09 mile more than the path recommended by *RecDist*, and she needs 0.32 minutes longer than *RecDur*. It also indicates that the average speed of the path recommended by *RecCert* is only 0.16 mile/hour shorter than that of the average speed of the path suggested by *RecDur*. Hence, we conclude that there is not much penalty (on an average) in choosing a more certain path.



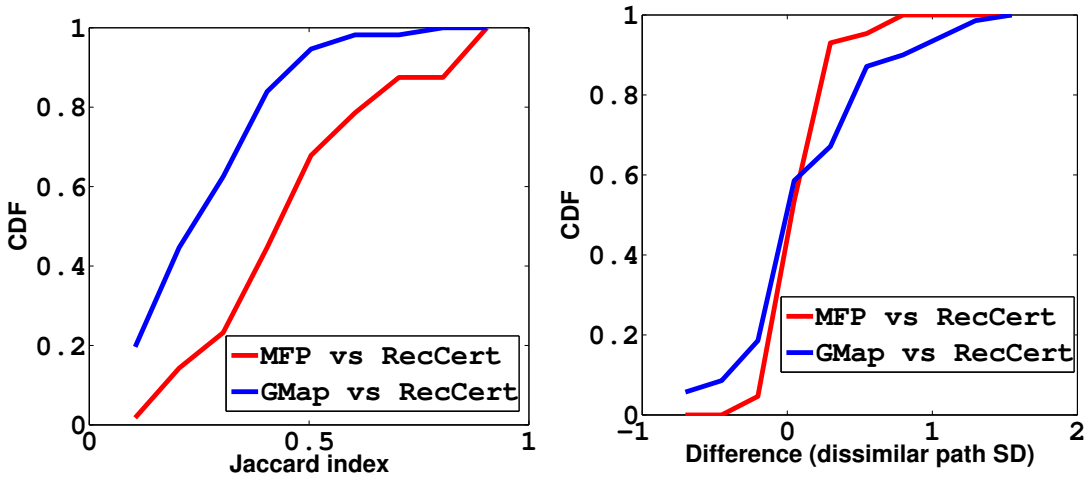
(a) (b)

Figure 3.7: Comparison of (a) CDF of path distance difference between *RecCert* and Waze, Google Maps (GMap), MFP (b) CDF of path duration difference between *RecCert* and Waze, Google Maps (GMap)

3.6.2 Comparison with Popular Route Recommender

In this section we compare our system with other existing commercial systems namely Google Maps, Waze and also time period-based most frequent path (MFP) [29] approach.

Baselines: Google Maps provides route planning and real time traffic information. Through Waze, drivers connect to one another and share real time traffic and road info with fellow drivers and help each other to improve their driving experience. Both Google Maps and Waze don't publicly provide their route recommendation algorithm. MFP calculates the



(a) (b)

Figure 3.8: (a)Path similarity and (b) Dissimilar path variance comparison between *RecCert*, Google Maps (GMap) and MFP

path frequency in terms of link frequency and selects the most frequent one.

Experimental setup: To measure the performance of our system against the baselines we choose Grand Central Terminal (GCT) in Manhattan as our destination and choose top 60 OD pairs from our trip data which have taken the longest time for reaching GCT between 4:00-5:00 pm in 2014. For each such OD pairs we collected recommended route information by GoogleMaps (via API) and Waze for a weekday, setting taxi departure time between same 4:00-5:00 pm. Note that the baseline commercial systems allow us to collect recommendation information for current and future date only. Next we find the route recommendation information by *RecCert*, *RecDur* and MFP from history data for same test OD pairs, date and time but for year 2014. For Waze we use the web interface (no API available) and could get only distance and duration information except the route.

Comparison of distance and duration: In table 3.6 we found, there is not much penalty in choosing a more certain path using *RecCert*. Hence in figure 3.7, we comparison our *RecCert* algorithms with three baselines Waze, Google Maps and MFP. Figure 3.7-(a) shows that all the methods predict paths with similar distances as their difference is close to zero, while Figure 3.7-(b) shows that Google Maps recommends paths with highest travel time, followed by MFP (which is very close to *RecCert*), and Waze recommends the fastest paths.

Comparison of non-overlapping path segments: Figure 3.8-(a) shows CDF of Jaccard index between paths given by baselines (Google Maps and MFP) and *RecCert*. Here the Jaccard similarity is calculated by viewing paths as sets of links. We see that all the three methods recommend very different paths, thus reaffirming our belief that in many scenarios there are many different "good" routes between origin and destination. Moreover, we see that paths given by MFP are more similar to those by *RecCert* than those given by Google Maps. Surprisingly, as shown in Figure 3.8-(b), difference in standard deviation between paths recommended by the three systems is low (despite the links being different). We see that Google Maps recommends slightly more uncertain paths than MFP and *RecCert*. Another interesting observation throughout this section is that MFP and *RecCert* behave very similarly while recommending paths in terms of travel time, distance and standard deviation. One might conjecture that taxi drivers in Manhattan intuitively follow the route of least variability.

3.7 Conclusion

The primary contribution of this work is to demonstrate a simple yet effective framework, to calculate mean travel time as well as variance for each link in a map, using **LSEC**. This is achieved on a large historical taxi trip dataset, which contains only the *end point information* for each trip. The work shows that using a road network, and a simple concept of non-ambiguous shortest paths, we can reconstruct routes and estimate travel time and variance on a city-wide scale, which none of the existing research systems do. We also demonstrate that the data derived can be effectively used to recommend paths, and thus introduce the concept of certainty based route recommendation. Our performance vis--vis commercial systems is also comparable; note that the performance of systems like Google Maps and Waze is based on real-time data from individual users, while we work on historical (2014) data. Interestingly, we find that the routes followed by our certainty based scheme are different from what Google Maps is suggesting and are found to be less prone to any on-road uncertainty. We are perhaps being able to capture alternate paths due to the scalability of the system, whereby we are able to capture an entire view of the city – however, this needs further investigation.

The percentage of links whose mean and standard deviation could be calculated, is around 80% and 50% respectively. We believe the coverage would increase further if we work with more data. A simple way to enhance data would be to consider data from all seven years (2009 - 2015) together. To use the data effectively, we may have to, however, perform some minor adjustments due to shift in travel time distribution over years, which would be an immediate future research agenda. Finally, we plan to make the annotated (with derived route information) NYC taxi data public for future research endeavours.

Chapter 4

Link Covariance Estimation

4.1 Introduction

In any urban setting, congestion on road is almost an unavoidable feature. Moreover, spread of congestion follows a complex path - congestion at one place may quickly affect nearby and sometimes even distant zones, causing inconvenience to people. Hence an important research question is to identify the range of influence of congestion occurring at an area. More specifically, we may want to know the correlation in traffic pattern between any two stretches of roads (links) within a city - that is, we would like to derive the covariance matrix of a city road network. Understanding the travel time correlation between the links has its importance with applications ranging from detection of traffic behaviour and anomalies at different places and times [35], assessing and improving overall efficiency of transportation systems [44].

Studies on link travel time corelation assume either correlations in link travel time are restricted to adjacent links [33, 34] or within a local area [11]. Although, in general, this may be true but a few instances of long-range correlation normally exist in a city-wide setting, identification of which may be important for various city-planning related applications. Hence, a model should not be constrained by the locality of correlation assumption. Another important criticism regarding the above mentioned models is the inherent assump-

tion that the information regarding link travel time is available - which is not true in practice due to several privacy and related concerns. However, path level travel information from city cab hiring services like Uber[1] are increasingly getting available. In line with that spirit, in this work, we use the path attributed taxi trip data from New York City[2] for the year 2014. The dataset provides the taxi trajectory information as well as the total trip time, but no information regarding the link travel time.

We estimate the average travel time for each individual links from this path attributed taxi trip data set and then derive the variance in travel time of those links as well as the covariance between any two links. Here the challenge is to infer the link covariances from observed path covariances. We solve this problem under the assumption that path travel times are linear combinations of link travel times and link travel times are distributed according to multivariate Normal distribution. Unlike existing methods, our method does not assume any side information or restriction on the link covariance structure. We call this Dependent Link Variance (DLV) method. In practice, however, there is a need to identify the most correlated links rather than having information about all possible correlated pair. To locate the most correlated links, we impose a sparsity constraint on the dependent link variance model using the sparse inverse covariance estimation method following [17]. A positive correlation between two links indicates that the traffic change (increase or decrease) in one link is directly proportional to the traffic change in the other link and vice versa.

The solution of this problem, helps us to achieve the following objectives: First, our DLV model is able to derive the link covariance resulting in an impressive improvement in accuracy (12%) of estimation of path variance vis-a-vis a model where link travel time is thought to be independent. Second, we have been able to estimate travel time correlations between distant links using our DLV model. Our method has been able to identify several pairs of links far apart in distance (~ 1.5 mile) but are highly correlated in terms of travel time. Third, through sparsification, the set of most positively correlated links identified lie within an average proximity of ~ 0.14 mile from the most traffic prone locations in the city – thus reaffirming the correctness of our correlation estimation framework. Fourth, we are successful in predicting the most affected paths during city events (e.g. parade events in Manhattan City namely: *Pulaski Parade*, *America's Parade* and *St. Patrick's Parade*),

[1] https://movement.uber.com/cities

[2] http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

using the correlation estimation in link travel time. The result shows that the Jaccard Similarity between the set of observed and estimated most affected path is $\sim 76.5\%$. The success of this model may help travellers to take necessary precaution while route planning during the time of the event.

4.2 Problem Formulation

Let $G = (V, E)$ denote the road network, where V is the set of all *nodes* in the road network, $E \subseteq V \times V$ is the set of all road *links* in the network. We assume a directed graph with $e = (v_1, v_2)$ implying that traffic can flow from v_1 to v_2. A path p is an ordered list of nodes (V_p) such that $(v_{i-1}, v_i) \in E, \forall i$. Equivalently, it can also be thought of as a list of links (E_p) such that if $e_{i-1} = (u_{i-1}, v_{i-1})$ and $e_i = (u_i, v_i)$, then $u_i = v_{i-1}, \forall i = 1, \ldots, M$, where M is the total number of links in the road network. The input trips dataset $\mathcal{D}$, is a collection of paths p_i along with trip travel time T_i, $\forall i = 1, \ldots, N$, where N is the total number of of trips. Note that a given path p_i may have multiple trips. Hence, we can compute the mean travel time $\bar{T}_p$ for a path p as $\bar{T}_p = \frac{1}{N_p} \sum_{i:p_i=p} T_i$, where N_p is the number of trips through path p. Similarly, we can compute the total variance S_p in travel time for path p as: $S_p = \frac{1}{N_p - 1} \sum_{i:p_i=p} (T_i - \bar{T}_p)^2$. Next, we describe our methods for covariance estimation in link travel time.

4.2.1 Link Travel Time Estimation

Let $X_k, k = 1, \ldots, M$ denote the random variables capturing travel times for the k^{th} link. Also, let T_p be the random variable denoting travel time for path p. We can write:

$$\sum_{k \in p} X_k = T_p + \eta \tag{4.1}$$

where, η is a Gaussian noise with mean ϵ close to 0 and variance σ^2. This is a simplified model, which ignores waiting time at the nodes. Taking expectation w.r.t. all random

variables X_k and noting that $x_k = \mathbb{E}[X_k]$ and $\bar{T}_p = \mathbb{E}_X[T_p]$, we formulate

$$\sum_{k \in p} x_k = \bar{T}_p + \epsilon \tag{4.2}$$

With this model, and the data for path travel times $\{\bar{T}_1, \ldots, \bar{T}_N\}$, we can write the above equations in a compact form as

$$A\boldsymbol{x} = \bar{\boldsymbol{T}} + \epsilon \mathbf{1} \tag{4.3}$$

where $\boldsymbol{x} = [x_1, \ldots, x_M]^T$, $\bar{\boldsymbol{T}} = \left[\bar{T}_1, \ldots, \bar{T}_N\right]^T$, and $\mathbf{1}$ is the vector of all ones. $A_{M \times N}$ is a coefficient matrix, where $A_{ik} = 1$ if i^{th} link is a part of k^{th} path, $A_{ik} = 0$ otherwise. Since we are interested in estimating x_k such that the mean $\epsilon \to 0$ in the Gaussian noise η, we use maximum likelihood paradigm to estimate the mean link travel times x_k with the constraint $x_k \geq 0, \forall k = 1, \ldots, M$ and the final estimation problem becomes:

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmin}} \parallel A\hat{\boldsymbol{x}} - \bar{\boldsymbol{T}} \parallel_2 \quad \text{s.t} \quad \boldsymbol{x} \geq 0 \tag{4.4}$$

This is the constrained least-squares estimation problem for estimating the mean.

4.2.2 Link Variance Estimation

The equation 4.1 can be written in a compact form as:

$$A\mathbf{X} = \mathbf{T} + \eta \mathbf{1} \tag{4.5}$$

where $\mathbf{T} = [T_1, \ldots, T_N]^T$. Taking co-variance on both sides of equation 4.5, and assuming that the noise η is independent of $\mathbf{T}$ and independent of each other, we get:

$$A\Sigma_X A^T = \Sigma_T + \sigma^2 \mathbf{I} \tag{4.6}$$

where, Σ_X and Σ_T are covariances of $\mathbf{X}$ and $\mathbf{T}$, respectively and σ^2 is the variance in noise.

First we make a simplifying assumption that all components of $\mathbf{X}$ and $\mathbf{T}$ are uncorrelated. Hence, $\Sigma_X = diag(\mathbf{s})$ and $\Sigma_T = diag(\mathbf{S})$, where $\boldsymbol{s} = [s_1, \ldots, s_M]^T$ and $\boldsymbol{S} = [S_1, \ldots, S_N]^T$ are variances of $\mathbf{X}$ and $\mathbf{T}$, respectively. The above equation can be

written as (since $(A\Sigma_X A^T)_{i,i} = \sum_{k=1}^{M} s_k A_{i,k}^2$ and $A_{i,k} = A_{i,k}^2$) :

$$A\boldsymbol{s} = \boldsymbol{S} + \sigma^2 \mathbf{1} \tag{4.7}$$

Since we are interested in finding $\mathbf{s}$ for the minimum variance in the noise, i.e. minimum σ^2, we get the following non-negative least-squared problem:

$$\boldsymbol{s}^* = \underset{\boldsymbol{s}}{\operatorname{argmin}} \parallel A\hat{\boldsymbol{s}} - \boldsymbol{S} \parallel_2 \quad \text{s.t} \quad \boldsymbol{s} \geq 0 \tag{4.8}$$

Note that the estimation problem is same as the link travel time estimation, except that the target value is path variance instead of path travel time. We call this method Independent Link Variance (ILV) estimation.

4.2.3 Covariance Estimation in Linear Models

Traditionally, covariance has been estimated from the direct measurements $\{x_1, \ldots, x_n\}$ using the scatter matrix $S = \sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T$. The covariance matrix becomes $\Sigma = \frac{1}{n-1}S$, when the mean or distribution is unknown, and $\Sigma = \frac{1}{n}S$ when x_i are assumed to be Gaussian. For high-dimensional covariance matrices, a common problem is low rank of the estimated covariance matrix which can be tackled using shrinkage estimators proposed in [25, 45], and many subsequent works (e.g. multi-target shrinkage [23], and alternative target matrices [54], etc.). In our case, we have a linear map T of the original random vector (x) of link travel times. Note that in our case, both x and T are high dimensional, ~ 2200 and ~ 22000 respectively. To the best of our knowledge, the problem of estimating covariance matrix of x, Σ_x from measurements of T has not been tackled before.

Given a Σ_T for the path covariance matrix, one can estimate the link covariance Σ_X using a least squares formulation based on equation 4.6, as $\sigma^2 \to 0$:

$$\begin{aligned} \underset{\Sigma_X}{\text{minimize}} \quad & \|A(\Sigma_X)A^T - \Sigma_T\|_F^2 \\ \text{subject to} \quad & \Sigma_X \succeq 0 \end{aligned} \tag{4.9}$$

This problem is an instance of convex semi-definite program (SDP) [7], where the constraint ensures semi-definiteness of the link covariance matrix Σ_X. However, the semi-

definite optimization suffers extremely high time complexity. Current SDP solvers utilizing interior-point methods scale as O($n^{4.5}$) or worse [21,47] where n is the number of rows in semi definite matrix. In order to circumvent the computational difficulties, we make a simplifying assumption that X follows a multivariate Gaussian distribution. Hence, we have the following model:

$$\begin{aligned} X &\sim \mathcal{N}(\bar{x}, \Sigma_x) \\ T|x &= Ax \end{aligned} \tag{4.10}$$

Marginalizing over x, we have:

$$P(T|A, \bar{x}, \Sigma_x, \sigma) = \mathcal{N}(A\bar{x}, A\Sigma_x A^T) \tag{4.11}$$

Note that the variance of path travel times T in this model turns out to be the same as the previous model (Eq. 4.6), except the additional variance ($\sigma^2 I$) introduced due to randomness while combining the link covariances, which is eliminated in the current model (Eq. 4.10). For this model, we can write the log-likelihood function given a trips dataset $\{T_1, \ldots, T_n\}$ as:

$$\begin{aligned} \mathcal{L}(\Sigma_x) &= \sum_{i=1}^{n} (T_i - \bar{T})^T (A\Sigma_x A^T)^{-1} (T_i - \bar{T}) \\ &\quad - n \ln(|A\Sigma_x A^T|) + \text{const.} \\ &= Tr(S_T (A\Sigma_x A^T)^{-1}) - n \ln(|A\Sigma_x A^T|) + \text{const.} \end{aligned}$$

where, $S_T = \sum_{i=1}^{n} (T_i - \bar{T})(T_i - \bar{T})^T$ is the scatter matrix for path travel time and $Tr(.)$ is the trace function. Also we can write $\Sigma_T = \frac{1}{n} S_T$. Next we take the derivative of $\mathcal{L}(\Sigma_x)$

with respect to Σ_x and set it to zero. The steps are as follows:

$$\begin{aligned}
\frac{d}{d\Sigma_x}Tr(S_T(A\Sigma_x A^T)^{-1}) - \frac{d}{d\Sigma_x}n\ln(|A\Sigma_x A^T|) &= 0 \\
\frac{d}{d\Sigma_x}Tr(\frac{1}{n}S_T(A\Sigma_x A^T)^{-1}) - \frac{d}{d\Sigma_x}\ln(|A\Sigma_x A^T|) &= 0 \\
\frac{d}{d\Sigma_x}Tr(\Sigma_T(A\Sigma_x A^T)^{-1}) - \frac{d}{d\Sigma_x}\ln(|A\Sigma_x A^T|) &= 0 \\
(\Sigma_x^{-1}A^{-1}\Sigma_T A^{-T}\Sigma_x^{-1})^T - A^T(A\Sigma_x A^T)^{-1}A &= 0 \\
(\Sigma_x^{-1}A^{-1}\Sigma_T A^{-T}\Sigma_x^{-1})^T &= A^T(A\Sigma_x A^T)^{-1}A \\
\Sigma_x^{-1}A^{-1}\Sigma_T A^{-T}\Sigma_x^{-1} &= A^T(A\Sigma_x A^T)^{-T}A \\
&= A^T(A\Sigma_x A^T)^{-T}A \\
&= A^T(A\Sigma_x A^T)^{-1}A \\
&= A^T A^{-T}\Sigma_x^{-1}A^{-1}A \\
&= (A^{-1}A)^{-T}\Sigma_x^{-1}(A^{-1}A) \\
&= I^{-T}(\Sigma_x)^{-1}I \\
\Sigma_x^{-1}A^{-1}\Sigma_T A^{-T}\Sigma_x^{-1} &= \Sigma_x^{-1} \\
\Sigma_x^{-1}A^{-1}\Sigma_T A^{-T}(\Sigma_x^{-1}\Sigma_x) &= (\Sigma_x^{-1}\Sigma_x) \\
\Sigma_x^{-1}A^{-1}\Sigma_T A^{-T}I &= I \\
(\Sigma_x\Sigma_x^{-1})A^{-1}\Sigma_T A^{-T}I &= \Sigma_x I \\
IA^{-1}\Sigma_T A^{-T}I &= \Sigma_x I \\
\Sigma_x &= A^{-1}\Sigma_T A^{-T}
\end{aligned}$$

Therefore, we get

$$\Sigma_x = A^{-1}\Sigma_T A^{-T} \tag{4.12}$$

However in 4.12, we consider that A is invertible, i.e., A is a square matrix. But in reality our A is $N \times M$ matrix, where N is total number of paths and M is total number of links, and so A is not a square matrix. But $A^T A$ is a square matrix and by the properties of square matrix, it should be invertible. So next we try to modify equation 4.12, so as to replace A^{-1}

with $(A^TA)^{-1}$ and get the final equation. The steps for this modification is as follows:

$$\begin{aligned}\Sigma_x &= A^{-1}\Sigma_T A^{-T} \\ &= A^{-1}I^T\Sigma_T I A^{-T} \\ &= A^{-1}(AA^{-1})^T\Sigma_T(AA^{-1})A^{-T} \\ &= A^{-1}(A^{-T}A^T)\Sigma_T(AA^{-1})A^{-T} \\ &= (A^{-1}A^{-T})A^T\Sigma_T A(A^{-1}A^{-T}) \\ &= (A^TA)^{-1}A^T\Sigma_T A(A^TA)^{-1}\end{aligned}$$

So we can finally write the modified equation as

$$\Sigma_x \quad = (A^TA)^{-1}A^T\Sigma_T A(A^TA)^{-1} \tag{4.13}$$

where there is no A^{-1} terms and equation 4.13 can easily be applied irrespective of whether A is invertible or not. Therefore, finally we can write:

$$\Sigma_X^* = \arg\min_{\Sigma_x} \mathcal{L}(\Sigma_x) = (A^TA)^{-1}A^T\Sigma_T A(A^TA)^{-1} \tag{4.14}$$

Here, we assume that the matrix A^TA is invertible. Also, note that the matrix A^TA stores the counts of common paths between links, i.e. $(A^TA)_{i,j}$ = number of common paths between links i and j. Hence, it is expected that A^TA is a sparse matrix, and easily invertible. Also to make sure that the A^TA is a full rank matrix for the real data, we add small λ (10^{-3}) to each diagonal elements of A^TA and thus our final equation becomes:

$$\Sigma_X = (A^TA + \lambda I)^{-1}A^T\Sigma_T A(A^TA + \lambda I)^{-1} \tag{4.15}$$

We call this our Dependent Link Variance (DLV) model.

Estimation of Path Covariance Matrix: In the above derivation, we assume that each vector T_i carries a simultaneous observation of path travel times for all available paths. However, in practise, the number of such simultaneous observations are extremely limited. In general one gets path travel times for a subset of paths at any given time window. Estimation covariance matrix with missing data can be performed in many ways [27]. However, due to extremely high dimensionality of the path covariance matrix ($22K \times 22K$), we con-

struct paired data between i^{th} and j^{th} paths as the pairs of path travel times whose starting times are no further than a threshold (say 30 minutes). Let there be K such paired data-points, $(T_{ik}, T_{jk}), k = 1, \ldots, K$ for the path-pair (i, j). We compute the $(i, j)^{th}$ entry of path covariance matrix $\Sigma_T(i, j) = \frac{1}{K}\sum_{k=1}^{K}(T_{ik} - \bar{T_i})(T_{jk} - \bar{T_j})$. While this procedure is not guaranteed to yield a positive semi-definite matrix, in practise find the resultant matrix to be PSD.

4.2.4 Sparse Inverse Covariance Estimation:

While the full covariance matrix can be useful in finding correlation between paths, the problem of finding most important correlations between links becomes important from the transport planning point of view. Also, many correlations are transitive, i.e. A is correlated to B and B is correlated to C implies A is correlated to C. This type of correlation can be avoided by measuring partial correlation, which is given (in the Gaussian case) by the *Precision* or inverse correlations matrix. The problem of estimating sparse inverse covariance matrix has been addressed in various domains, including computational biology [17]. The sparse inverse covariance matrix W, can be estimated as:

$$\min_{W} \log |W| - Tr(W\Sigma_X) - \gamma\|W\|_1 \tag{4.16}$$

where Σ_X is our estimated non-sparse covariance matrix. Once we estimate the link covariance matrix Σ_X, we get the estimated path covariance matrix as $\hat{\Sigma_T} = A\Sigma_X A^T$.

4.3 Experimental Setup

Dataset: The path attributed dataset has been constructed with the help of New York City taxi trip dataset[3] provided by the New York City Taxi and Limousine Commission for the year 2014 and the OpenStreetmap (OSM)[4] data as discussed in section 3.2.2. However here we consider an one month path attributed taxi trip data starting from 18^{th} June 2014 to 18^{th}

[3] http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

[4] http://www.openstreetmap.org

July 2014 around Manhattan mid town (figure 4.4.1). During this one month period, there were 800626 taxi trips, covering 21132 taxi trajectories, spanning over 2112 road links (edges) in mid town Manhattan.

Baseline: To compare the performance of our Dependent Link Variance (DLV) (section 4.2.3), we use the Independent Link Variance (ILV) model (section 4.2.2) as the baseline method where link travel time is thought to be independent.

Procedure and Metric: For the experiment, first we randomly split the trip dataset into 80% training set and 20% testing set. Next using the training set, for a pair of paths, we consider a *time bucket* (TW) of 2 hours and consider all the trips in those paths whose trip start time is within 2 hours. Thus for each pair of paths we build this *pair trip data* and calculate the correlation in the path travel time and build the path covariance matrix Σ_T (section 4.2.3). Using this training set data, we learn our model to estimate the link covariance Σ_X using different regularization parameter λ. Once we estimate the covariance Σ_X in link travel time using DLV, we measure it's performance two ways. First we compare the performance of DLV for different values of λ. For this we calculate the estimated path covariance $\hat{\Sigma_T}$ for the 20% test data using our learned Σ_X and compare it to the observed path covariance Σ_T in the test data. For measuring the performance of our model, we use two metrics, MAPE – mean absolute percentage error and RMSE – root mean square error. We use these two metrics for measuring the performance between the observed and estimated path covariance respectively. Second we compare our DLV method with the baseline ILV method. However in ILV links are uncorrelated. So we have only the estimated link variance for the baseline ILV method. Therefore we compare the performance between DLV and ILV as follows. For this, we calculate the estimated path variance $\hat{V_p}$ for the 20% test data using both DLV and ILV method and compare it to the observed path variance V_p by calculating the MAPE and RMSE between them.

Lambda (λ)	10	1	10^{-1}	10^{-2}	10^{-3}	10^{-4}
MAPE (%)	25.97	24.05	23.95	23.51	**23.44**	23.44
RMSE (min)	2.51	2.00	1.87	1.85	**1.86**	1.86

Table 4.1: Performance of MAPE and RMSE with different values of λ for Dependent Link Variance Model. Here we compare between observed and estimated path covariance.

Method	MAPE (%)	RMSE (min)
ILV	26.03	3.30
DLV	22.84	1.73

Table 4.2: Comparison of MAPE and RMSE between Dependent Link Variance (DLV) and baseline method Independent Link Variance (ILV) model. Here we compare between observed and estimated path covariance.

4.4 Result

In this section we measure the performance of our link covariance estimation task. To compare our result, First: Using both the ILV and DLV models, we find the estimated path variance and compare with the observed path variance in the test data. Second: We implement the *sparse inverse covariance* estimation as described in section 4.2.1 (equation 4.16) using our estimated link covariance Σ_X (by DLV) and figure out the most correlated links of the city. We try to justify our prediction of the most positively correlated links, by plotting them on the global maps, which shows that those predicted links are around the real world traffic prone zone in mid town Manhattan.

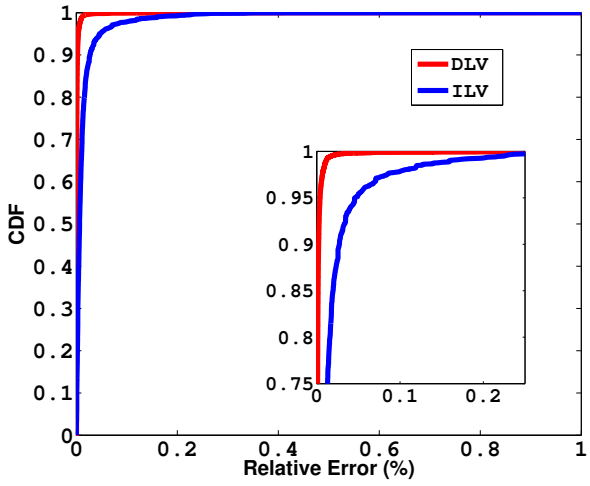


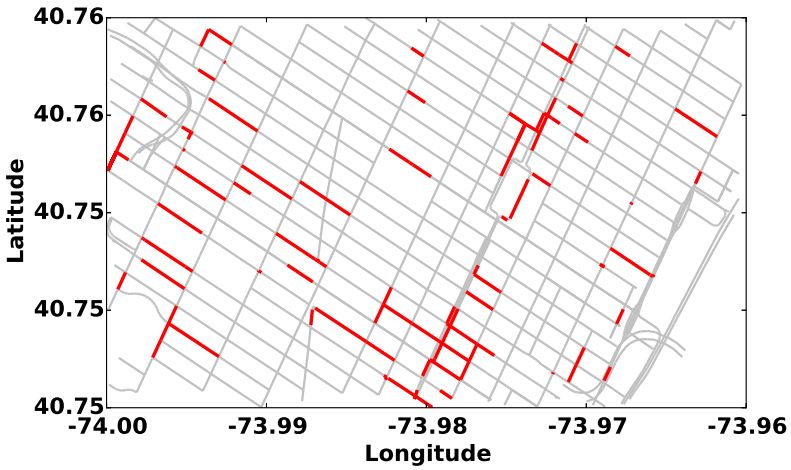
Figure 4.1: *CDF* of relative error (%) for Dependent Link Variance (DLV) ($\lambda = 10^{-3}$) and Independent Link Variance (ILV) model

4.4.1 Baseline Comparison

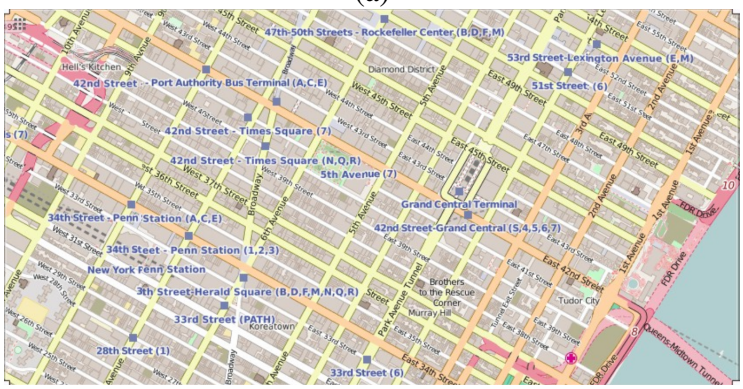
For DLV model, we run our algorithm (equation 4.15) on the path attributed spatio-temporal taxi trip data spanning over mid town Manhattan for estimating the link covariance matrix Σ_X for different λ. Next using the link covariance matrix we calculate the estimated path covariance over the test set and calculate the MAPE and RMSE between the observed and estimated path variances. In table 4.1 we show the performance of our DLV model for different regularization parameter λ. We see that for $\lambda = 10^{-3}$, the MAPE and RMSE achieves minimum value. The MAPE value is 23.44% and RMSE in 1.86 minute. Next we run our baseline ILV method (equation 4.8), on the same spatio temporal dataset. Note that, due to the assumption in ILV that links are uncorrelated, the ILV method has been able to estimate link variance only. Finally using these link variances we estimate the path variance for the same test data and calculate the MAPE and RMSE between the observed and estimated path variances. Table 4.2 shows that by using DLV, our MAPE has been decreased from 26.03% (by ILV) to 22.84% (improvement 12%) and RMSE has gone down from 3.30 minute (by ILV) to 1.73 minute (improvement $\sim 47\%$). The reason for such better performance in DLV model is due to the assumption that links are correlated and thus bringing the estimated path variance more closer to the observed path variance as compared to the baseline uncorrelated ILV model. For a more deeper understanding, we do a CDF analysis for the absolute relative error between the observed and estimated path on the test set data, using both DLV and ILV model. In figure 4.1, we see that, using our DLV method, $\sim 99\%$ of the estimated path variances are within an error range of 5%. But for baseline method ILV, $\sim 95\%$ of the estimated path variances are within the same error range. Which clearly depicts the improvement of DLV method over the ILV method.

4.4.2 Understanding the Link Correlation Network

The estimated Σ_X is not sparse and shows correlation between every pair of links – resulting in a fully connected link correlation graph. However, an utility of the model is to figure out the most correlated link set due to city traffic. For this purpose we apply the sparse inverse covariance estimation [17] (equation 4.16, section 4.2.1) on the estimated link covariance (using DLV method) and derive the sparse inverse link covariance network. The



(a)



(b)

Figure 4.2: (a) Prediction of most crowded road links in the mid town Manhattan (b) The annotated map of the experimental region.

experiment has been done on the same spatio-temporal taxi trip data (section 4.3) around mid town Manhattan City (figure 4.2). The links which are positively correlated in the network form the candidate correlated set. To measure the goodness of the estimation we posit that a good estimation would choose links which are within a close proximity of the actual crowded/traffic prone locations in the experimental region (mid town Manhattan).

Most Affected Links – Ground Truth Preparation: Unfortunately there is no direct traffic survey readily available which mentions about the most crowded/traffic prone locations in mid town Manhattan. So we manually prepare this ground truth information from different news article and QA site from Internet. The sources are namely: *15 Areas in NYC*

Most Traffic Prone Locations
Herald Square , W 35th Street
Times Square, 7th avenue & 46th - 47th street
Madison square , 34th street 7th avenue
Rockerfeller Center
8th Avenue & W 42nd St
Penn Station
7th Avenue & W 42nd St
Grand Central Station
7th Avenue & W 34th St
Port Authority
Lexington Avenue & E 42nd St
5th Avenue and 23rd Street

Table 4.3: Ground Truth data for the most busiest locations inside the experimental region – mid town Manhattan.

Predicted Locations (most crowded)	Nearest Observed Locations (most crowded)	Distance (mile)
GCT, 89, East 42nd Street	self	**0.0**
Citi Bike - E 47 St	GCT	0.3
230, Park Avenue,	GCT	0.2
Stern College Yeshiva University	GCT	**0.0**
481, 8th Avenue	Penn Station	0.1
W 34th Street,	Penn Station	0.2
33 W 42 St / 7 Ave	Port Authority	0.4
W 42 St / 7 Av, West 42nd Street,	7th Avenue & W 42nd St	**0.0**
336, 8th Avenue, Chelsea,	Madison Square Garden	0.2
Herald Square	self	**0.0**

Table 4.4: Comparison between the Predicted and Observed most crowded locations

that are Crowded 24/7[5], *What are the most crowded pedestrian areas in NY?*[6], *What is the*

[5]http://housely.com/15-areas-nyc-crowded-247/

[6]http://bit.ly/2fD9wdH

busiest street in manhattan?[7], *Busiest Intersections in NYC*[8], etc. Based on the information available in these websites, we prepare our ground truth for the most congestion prone locations inside our experimental region. The table 4.3 shows the list of most crowded/traffic prone region inside mid town Manhattan (figure 4.2). Some of these places are Grand Central Station (GCT), Penn Station, Port Authority, etc. The first two are train stations and the third is a Bus Terminal. Some of the other crowded places are RockerFellar Center, Times Square, Herald Square, 5^{th}, 7^{th}, 8^{th} avenue, Lexington avenue, etc.

Comparison With Ground Truth Links: After applying the sparse inverse covariance estimation [17] on our estimated Σ_X, we plot the correlated links in the global map. Figure 4.2-(a) shows the plot on the road network of the experimental region. The links in red are correlated to each other as obtained from sparse inverse covariance estimation. The links in grey colour denote the non correlated region. Also figure 4.2-(b) shows the road network with annotation of location names. The two figures together provide a visual clue to gauge the traffic prone locations of correlated links that our system has been able to predict.

Table 4.4 shows the performance of prediction with respect to its proximity to the most traffic prone locations. The first column shows the predicted traffic prone link locations while the second column denotes the nearest ground truth crowded/traffic prone links. which justify how close our result is to the actual congested locations. We see that our predicted most correlated link locations *GCT, 89, East 42nd Street*, *Herald Square* (table 4.4) directly map with the ground truth traffic prone locations (table 4.3) at *Grand Central Station* (GCT) and *Herald Square*. We also see that link locations at *Citi Bike - E 47 St*, *230, Park Avenue*, *Stern College Yeshiva University* are very close to the observed crowded locations *Grand Central Terminal*, within a range of ~ 0.2 mile. We also see that *481, 8th Avenue*, *34th Street* are also near to the *Penn Station* – one of the busiest railway stations in Manhattan (within 0.3 mile). Also the *336, 8th Avenue, Chelsea* is actually close to *Madison Square Garden* (within ~ 0.3 mile), a place in Manhattan famous for organizing various events and programs. To conclude, it is observed that our sparse inverse covariance estimation method has actually predicted most correlated links within an average distance of ~ 0.14 mile from the most congested locations of the region.

[7] http://bit.ly/2kOvpUC
[8] http://bit.ly/2eHBekq

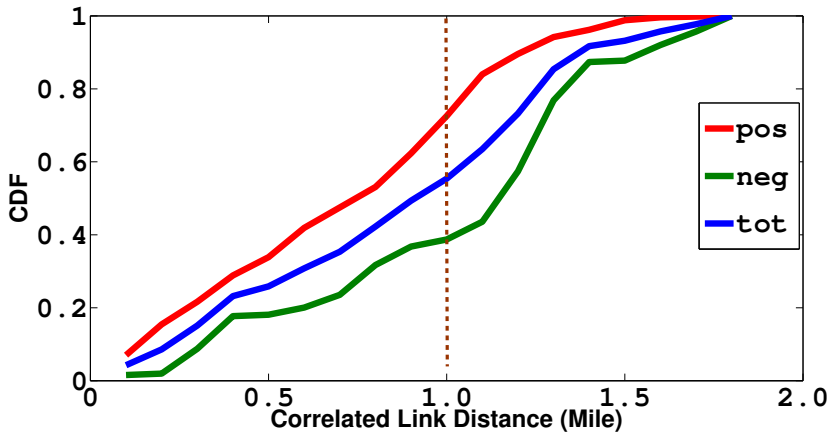


Figure 4.3: *CDF* of distance (mile) between correlated links for positive (pos), negative (neg) and all (tot) correlations

Understanding Long Distance Correlated Links: To understand, how good our method has been able to capture long distance correlations, we plot the CDF of inter link distance for the positive and negative correlated links separately. For two correlated links, we measure their intermediate distances by measuring the euclidean distance between the mid points of the two correlated links. Figure 4.3 shows, for $\sim 30\%$ positively correlated links (in red), the inter link distance is more than 1.0 mile. However, we see that $\sim 20\%$ negatively correlated links (in green) are less than 0.5 mile apart. Thus we see both long range as well short range link correlations exist. Therefore, the interesting phenomenon is that sometimes nearby links are actually negatively correlated, i.e, congestion at one point may lead to relatively free road link nearby!

Event Name	**Time**	**Location**	**JS** (%)
Pulaski Parade	5th Oct, 2014	From 39^{th} St to 56^{th} St	78.0
America's Parade	11th Nov, 2014	From 26^{th} St to 52^{nd} St	76.76
St. Patrick's Parade	17th Mar, 2014	From 44^{th} St to 79^{th} St	74.75

Table 4.5: Parade event details and the Jaccard Similarity (JS) between the estimated and ground truth most affected paths

4.5 Prediction of Most Affected Paths

We validate our estimation of link covariance by predicting the most affected paths during localized city events. We obtain the ground truth list of the most affected paths during those events from our path attributed trip data. Also using the past trip data before the events, we estimate the covariance in link travel time in the locality of the events and estimate the path variance of all the paths which has been traversed during the event. Finally we measure the similarity between these two lists to confirm the success of the prediction of the most affected paths.

Experimental Procedure: We choose three *parade events* namely: *Pulaski Day Parade*, *America's Parade* and *St. Patrick's Day Parade*[9] in 2014. The table 4.5 shows the time and venue of these events. First we find out the list of most affected paths from our path attributed trip data during these parades. We consider all the paths traversed within the locality of the parade venue. Next we calculate the average path travel time for all these paths from the past 2 weeks trips before the said event. We find the list of paths whose travel time has been deviated most from their average path travel time during the event and create the ground truth path list. Finally, using our DLV method, we calculate the covariance Σ_X in link travel time using all the trip data within the past 2 weeks trip data (excluding the event day) around the event location. Once we estimate the link covariance, we find the path variance of all the paths. At last, we take the top-k paths having highest predicted path variance and compare it with our ground truth list of most affected paths.

Performance: The third column in table 4.5 shows the performance of our prediction of the most affected paths. For each of the three parade events we predict the top 200 most affected paths and compare with the ground truth list of top 200 most affected paths. We see that for *Parade Pulaski*, the Jaccard Similarity is 78% between these two path lists. For *America's Parade* and *St Patrick Parade*, the similarity score is 76.76% and 74.75% respectively. These results show how good our DLV method is in estimating the link covariance,

[9] http://www.mustseenewyork.com/parades.html

which results in the good estimation of path variance during city events and helps to predict the most affected paths.

4.6 Conclusion

The primary contribution of this work is to demonstrate a simple yet generic framework, to calculate covariance in link travel time using Dependent Link Variance (DLV) model. This is achieved using the path attributed taxi trip dataset, which contains only the path and the trip travel time. Our model has been able to estimate link covariance, resulting in better estimation of path variance (improvement by 12%) as compared to the model with independent link assumption. Our sparse inverse link covariance estimation captures the most correlated links, which are in close vicinity of the congestion region. We are perhaps the first to identify the long distance link correlations and find out that $\sim 30\%$ positively correlated links are more than 1.0 mile apart.

Chapter 5

Mining Twitter and Taxi Data for Predicting Taxi Pickup Hotspots

5.1 Introduction

In large cities taxi services normally maintain hotspots; these are either city-specified taxi stands or certain spots which have evolved over time. The knowledge of hotspots helps daily/experienced commuters to quickly and easily avail taxi service. Such hotspots may get severely disturbed during a road blockage (e.g. closure due to maintenance) around that area whereby taxis dynamically build new hotspots. These new hotspots may be far away from older hotspots, hence causing inconvenience to the commuters as well as resulting in loss of business and time of taxi drivers. A service which can predict the new hotspots and accordingly suggest its users to the most convenient new hotspot would be of tremendous value to both experienced and new commuters.

In order for the scheme to be successful certain conditions need to be fulfilled. First, one needs an automatic technique to identify such events causing the change in hotspots. In order to predict (change in) hotspots, one needs historical information of hotspots, not only the location but also their strength - that is how many pick-up has been done from these hotspots. Luckily nowadays information of such events widely get publicized in so-

cial media. Therefore, we need to develop a scheme to identify them from the million other non-relevant information and accurately pinpoint the events, correct location. Secondly, various transportation datasets from different cities across the world like, Porto, Boston, Chicago, New York, etc., are made public either by the city authority or by taxi hiring company (e.g. Uber[1]) from which hotspot learning is becoming possible.

In this work, we concentrate our study on New York City. This is because both disruption information as well as taxi hotspot information are available for this region. Moreover, on studying, it is found that there is significant 'movement of hotspots' during any disturbance. The New York City Department of Transportation (twitter handle: NYC_DOT) regularly post tweets about various events in New York, out of which road, bridge closure is also announced. The New York City Taxi and Limousine Commission routinely releases information regarding trips of yellow taxi, their pick-up and drop-off points from which one can calculate the hotspots in the city. Thirdly, the impact of such road closure activity on taxi-hotspot movement in New York is generally poignant (figure 5.1 illustrates an example). Figure 5.1-(a) shows a tweet posted by the NYC_DOT regarding an road closure incident at *Brooklyn Bridge* from 23^{rd} May, 2016 to 22^{nd} June, 2016 from 7 am to 5 pm. Figure 5.1-(b) shows the average taxi pickup counts around the activity location (purple triangle in figure) from the past two weeks taxi trip data before the road closure incident takes place. The Regular Taxi Pickup Hotspot (RTPH) shows that the average pickup count around the location, before the road blocking activity occurs, varies between 0-75. The figure 5.1-(c) shows the Event Specific Taxi Pickup Hotspots (ESTPH). It depicts that, here also the total pickup count varies between 0-75 on the event day. But a close look at the two picture shows the difference in heat-map and indicates in the change of taxi pickup hotspot. Two regular hotspots in the upper middle zone in figure 5.1-(b) have shifted right on the event day figure 5.1-(c). One would also find similar changes in the bottom right zone of figure 5.1-(b),(c). Also a regular hotspot in the extreme left middle zone in figure 5.1-(b) is vanished completely in figure 5.1-(c). This clearly supports our claim about the hotspot relocation.

Summing up, we present a two step methodology to tackle this problem of finding taxi pickup hotspots by analysing posts from Online Social Network (OSN). We have used two

[1]https://movement.uber.com/cities

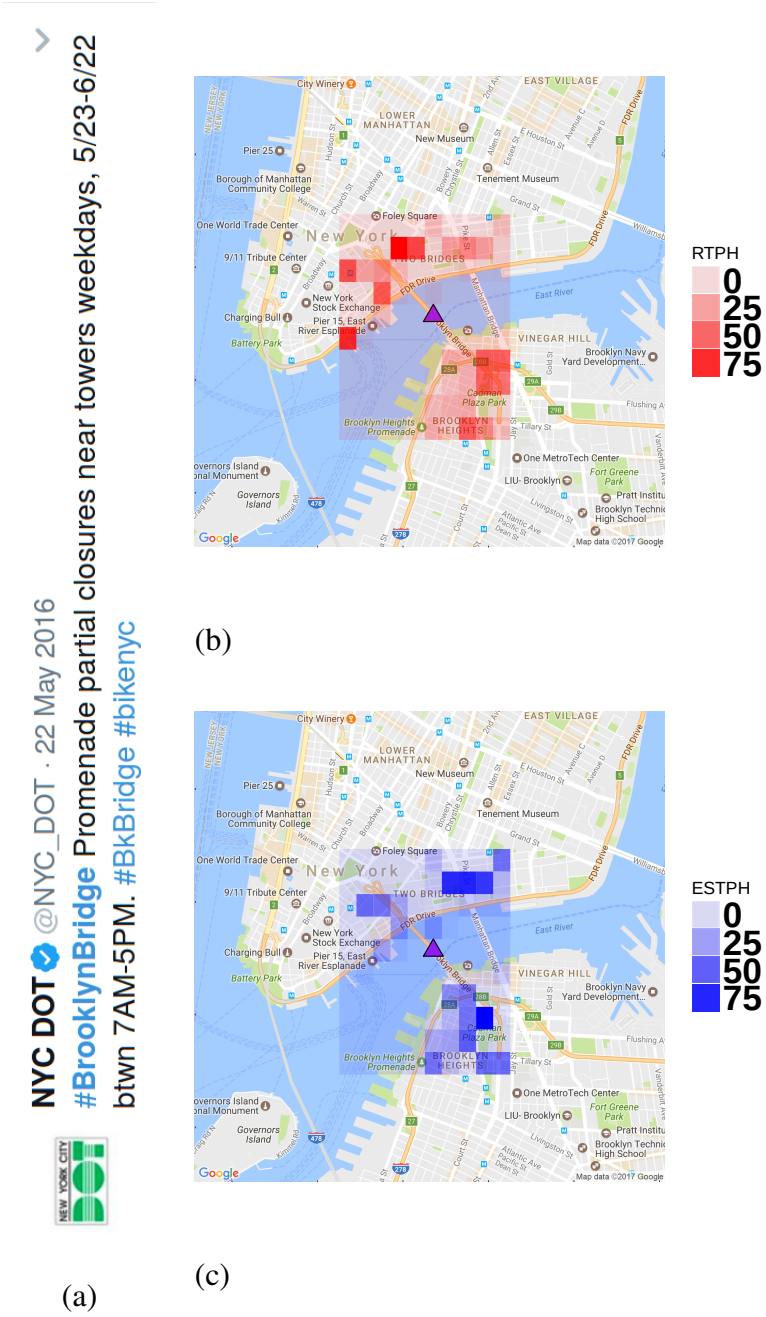


Figure 5.1: (a) Tweet notifying lane closure activity in Brooklyn Bridge. Heatmap showing (b) Regular Taxi Pickup Hotspot (RTPH) before and (c) Event Specific Taxi Pickup Hotspot (ESTPH) after road closure activity around Brooklyn Bridge (purple triangle).

datasets: (1) the tweets posted by New York City Department of Transportation (twitter handle: NYC_DOT[2]) for the year 2015 for inferring the social cue and (2) the New York

[2]https://twitter.com/NYC_DOT

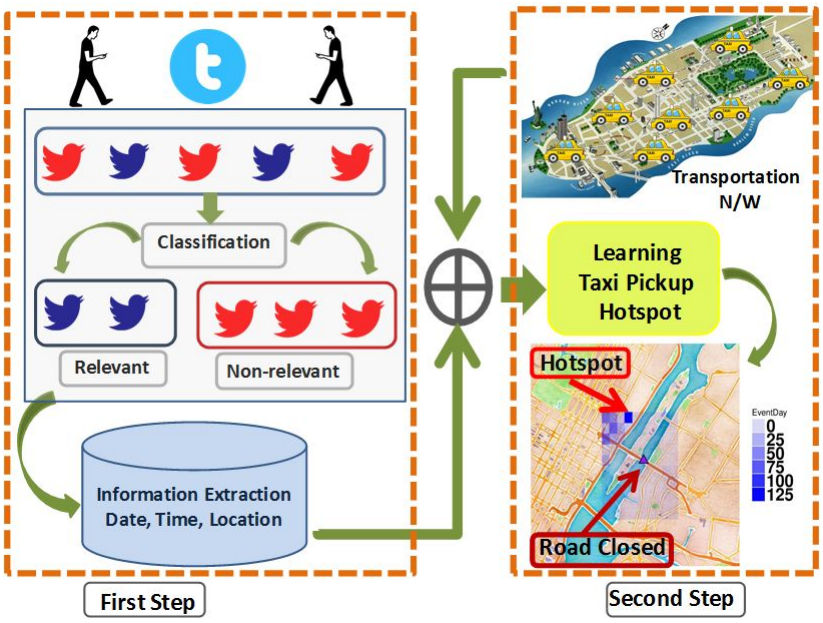


Figure 5.2: Two-step process architecture showing relevant information extraction followed by learning taxi pickup hotspot

City Taxi cab data[3] for the year 2015. Our two step process includes two broad task: (a) Natural Language Processing (NLP) task for information extraction and (b) Map-based task for hotspot prediction during road closure activity using the extracted information. Figure 5.2 depicts the overall model architecture of our two-step process. In the first step, we present method for NLP tasks, classify eventful tweets, followed by extraction of meaningful information on imminent traffic disturbances by using the notification tweet posted by NYC_DOT. After comparing the road closure incidents between, the year 2015 and 2016, we observe that among all the road closure locations, $\sim 40.47\%$ (section 5.2.2) event locations are repeated across years. Second we go for Map-based task where using the extracted information, we try to predict the taxi pickup hotspot around the activity location using the New York City Taxi cab data.

Finally, we summarize the key points of our works as follows:

- We show the method to learn the taxi pickup hotspots combining the online and offline data source (section 5.3, 5.5) using a two step process.

[3]http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

Location	Birth of NH (%)	Death of OH (%)	minimum shift (mile)
Battery Park	33	4	0.104
Brooklyn Bridge	0	11	0.105
Manhattan Bridge	82	0	0.000
Williamsburg Bridge	0	37	0.103
WillAve Bridge	0	0	0.000
QueensBoro Bridge	0	41	0.102
Roosevelt Bridge	11	19	0.105

Table 5.1: Hotspot relocation details showing birth of new hotspot (NH) and death of old hotspot (OH) in %. It also shows the minimum shift (in mile) of old hotspot to new hotspot.

- Our approach is successfully classifying the relevant and non-relevant traffic tweet with an average accuracy of ∼**94.55%**, precision ∼**0.945** and recall ∼**0.945** (section 5.4.1).
- Following the classification, we have successfully extracted date-time and location information from $\sim 80\%$ of relevant tweet (section 5.4.2).
- In section 5.6 we show how fair we are doing in predicting the pickup hotspot. It shows that our average relative absolute error (RAE) is only ∼**13.96%** and root relative squared error is ∼**22.33%** for locations with higher number of road closure incidents.
- Finally we see that our predicted taxi pickup locations are within an average radius of only **0.011** mile from the original pickup hotspots (section 5.6.3).

5.2 Importance and Feasibility of the Problem

The problem of understanding the taxi pickup hotspot relocation, using road closure event notifications from social networking sites, has it's own importance. Because, by tackling this problem, the urban commuters as well as the taxi drivers will be greatly benefited.

5.2.1 Importance of the Problem

In urban travel life, there are certain places, from where taxi drivers pickup passengers regularly. We define these locations as regular taxi pickup hotspot (RTPH), if the pickup count is greater than a threshold, as seen in figure 5.1-(b). For this analysis, the minimum taxi pickup threshold is set to 3. All the regular passengers know about these hotspots from their experiences. But whenever any road closure incident occurs, the taxi drivers has to change their regular travel route. This results in the change of the RTPH. We have the following three general observations:

Death of Old Hotspot: Here, some of the old hotspots no longer remain as hotspots, as the passenger pickup count drops significantly in those places. In figure 5.1-(c), some of the old hotspots become obsolete, due to road closure activity at Brooklyn Bridge, compared to figure 5.1-(b). This trend is also occurring in other places of New York City. After analysing the tweets posted by *NYC_DOT* in section 5.3, table 5.1 shows that, QueensBoro Bridge and Williamsburg Bridge suffer from the highest death of old hotspots, 41% and 37% respectively.

Birth of New Hotspot: As a side effect of the road closure incident, new hotspots also appear in the locality, where previously no passenger pickup was done – birth of new hotspot. In figure 5.1-(c), some new event specific taxi pickup hotspots (ESTPH) have come up around Brooklyn Bridge, where previously no hotspots were there, as seen in figure 5.1-(b). This phenomenon is happening in other locations of New York City as well, where such road closing activity occurs. The first column of table 5.1 also states that Manhattan Bridge (82%) and Battery Park (33%) got the most number of new hotspots due to road closure events.

Hotspot Relocation: The third column of the table also shows the minimum shift (in mile) of an old hotspot to a new hotspot for each of these locations due to such road blocking activity. The minimum shift is reported as ~ 0.104 mile. This shows, at the arrival of road closure events, at least how far the hotspots move from its usual places.

Year	#Incidents	#Location
2015	367	67
2016	417	42

Table 5.2: Total number of incidents and closure locations for year 2015 and 2016

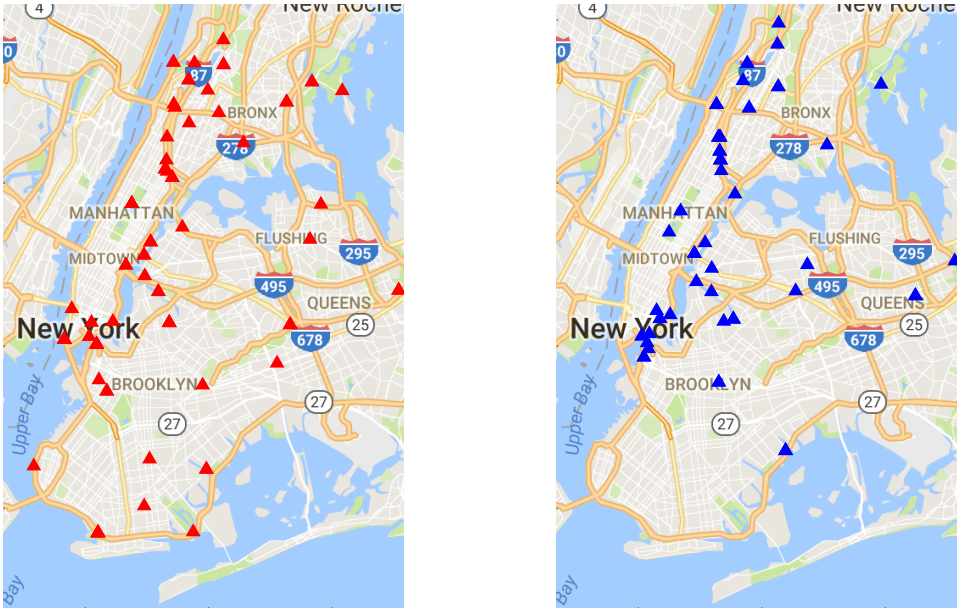
5.2.2 Feasibility of the Solution

Once the road activity location names are extracted from the tweets (section 5.3) for both the year 2015 and 2016, the following general observations have been made:

Total incidents: Table 5.2 reports 367 and 417 road closure incidents in New York City for year 2015 and 2016 respectively. These incidents are spanned over 67 locations in 2015 and 42 locations in 2016.

Affected region and overlap: To understand their geographical spreading, the coordinates of the incident locations are plotted on the global map. The figure 5.3 interprets that majority of the incident locations are densely spread over Manhattan City. But rest are sparsely located over Bronx, Brooklyn and Queens. The analysis also finds that there is 40.47% repetition in incident locations between these two years. This is even clearly visible in the figure 5.3. For Manhattan City, the geographic regions for the road closure incidents are very similar in 2015 and 2016. However for Bronx, Queens and Brooklyn, we see that the incidents occur in few new locations. This certainly tells that, these repeated road activities are mainly affecting Manhattan City.

Location Type: Next we focus on the type of locations which are the victim of these road closure incidents. The word-cloud in figure 5.4 shows that majority of the locations are mainly *bridges* in and around Manhattan City. Some of the major locations are *GrandStreetBridge* (GrandStBr), *BrooklynBridge* (BrooklynBr), *BatteryParkUnderPass*, *BeltParkway* (BeltPky), etc.



(a) (b)

Figure 5.3: Distribution of road closure event locations in New York city for year (a) 2015 (b) 2016. It shows major locations are distributed mainly in Manhattan city with good overlap across years.

(a) (b)

Figure 5.4: Word cloud showing which are the top locations suffering from most frequent road closure incidents in New York city for year (a) 2015 (b) 2016

Year	Duration	Count
2015	25/12/2014 to 12/12/2015	2981
2016	4/1/2016 to 12/12/2016	2966
	Total	5947

Table 5.3: Twitter dataset for road activity information inference for year 2015 and 2016.

5.3 Tweet Acquisition and Preprocessing

Our system for predicting hotspots for taxi-pickups has two major components:

- Automatically inferring information about road-closure incidents from social media (Twitter) data.
- Analysing and predicting traffic hotspots from traffic data.

In this section, we describe the first part of the two. Section 5.5 discuss about the second. Now we explain the source of social media data and describe classification of relevant tweets, while section 5.3.2 describes extraction of details of road closure from the relevant tweets.

5.3.1 Tweet data-source and classification

Twitter data-source: The NYC Department of Transportation tweets regularly about the traffic conditions in New York City from their official twitter handle NYC_DOT[4]. They post early notifications for various road closure activity across the city. We have crawled their twitter timeline for year 2015 and 2016 using twitter REST API[5]. Table 5.3 shows that 2015 dataset contains tweet feed from 25^{th} Dec 2014 to 12^{th} Dec 2015. For 2016, the dataset is from 4^{th} Jan 2016 to 12^{th} Dec 2016. The total number of tweets for 2015 and 2016 are 2981 and 2966 respectively.

[4]https://twitter.com/NYC_DOT
[5]https://dev.twitter.com/rest/public

Tweet classsification: We use standard techniques for classification of relevant tweets. We assume that $tf_i = \{f_{i1}, f_{i2}, \ldots, f_{iN}\}$ is the tweet-feature vector for i^{th} tweet, where f_{ik} denotes the k^{th} feature value for i^{th} tweet. We describe the features used in section 5.4.1. Let $C_i \in \{0, 1\}$ denote the class label capturing whether i^{th} tweet is relevant or not. Here by relevant, we mean one containing information about road closures. Our final objective is, to classify a given unseen tweet as relevant or not. In summary our input, outputs are as follows:

- **Input:** A fixed set of tuples having tweet-feature vector and corresponding class labels $[(tf_1, C_1), (tf_2, C_2), \ldots, (tf_M, C_M)]$
- **Output:** A binary classification model $\boldsymbol{CR} : tf \rightarrow C$ where $\boldsymbol{CR}$ classifies the tweet as relevant or not.

We solve this binary classification problem using four different machine learning techniques, (1) Support Vector Machines (SVM), (2) Naive Bayes, (3) Random Forests, and (4) Bagging, implemented in Weka [19]. We discuss the details implementation and results in section 5.4.

5.3.2 Inferring Road Closure Information

After identifying relevant tweets we proceed to extract three pieces of information about the particular road closure announcement: (1) Location, (2) Time and (3) Date.

Extracting Closure Location: To extract the location name, POS tagger is used for filtering the *singular* and *proper* noun. The *hashtags* are also considered, which explicitly contain the location names. First we normalize the tweet text to remove the Twitter specific encoding and obtain the tweets in English and proceed for information extraction. The details are as follows: (1) First, we perform parts of speech (POS) tagging for the tweet and select only the words having NNP[6] tag, giving us first set of candidate words. (2) All the

[6]proper and singular noun

hashtags are filtered which are used for mentioning the name of the event locations, giving us second set of candidate words. Combining these two candidate sets, our final set of potential candidates is generated for location information extraction. (3) Finally, we query the candidate words to find the list of global coordinates (latitude,longitude) which these words might represent using python *geopy* library and choose only those which fall inside New York City.

Extracting Time: The primary challenge while extracting *date*, *time* information from relevant tweet is the diverse ways in which people use *date*, *time* information in their tweets. First all the text, which are succeeded by am or pm, are extracted using the regular expression: $*(am|pm)*$. But a major challenge is to map the *time* information to the corresponding *date* information. This is critical for event happening over multiple days or multiple times during a day. To overcome the first challenge we use regular expression based rules to filter out *date*, *time* information from other numeric and alphanumeric strings. To overthrown the second bottleneck, we also mark the position of the occurrence of the *date*, *time* information and arrive at the mapping based on relative position of the *date* and *time*.

Extracting Date: To identify the *date* information we check for *absolute mention* and *relative mention* of the date. Example of *relative mention* will be *tomorrow*, *today* and the shorter form for these words. We use the regular expression

$$\text{“}tomorrow|tmrw|yesterday|tonight|today\text{”}$$

for this task. If a relative reference for *date* is found then the date for the event is calculated using the posting time of the tweet. We also take the help of *conjunction words*. These words help to identify the duration of the event and draws relation between different dates and time. Types of conjunction words used in our regular expression are

$$\text{“}through|thru|to|between|btn|\&|and|-\text{”}$$

. Finally, in the next section, we combine these extracted knowledge with the transportation data for predicting taxi pickup hotspot during road blocking events.

	KW	Date	Time	Day
2015	0.237	0.114	0.107	0.019
2016	0.262	0.235	0.301	0.040

Table 5.4: Information gain analysis for feature selection – *Key Words* (KW), *Date*, *Time* and *Day* for tweet classification

	Acc(%)	P	R	F-Score	ROC
SMO	92.82	0.925	0.928	0.921	0.767
RF	93.85	0.936	0.939	0.936	0.927
BG	**93.86**	**0.938**	**0.939**	**0.938**	**0.930**
NB	93.09	0.931	0.931	0.931	0.930

Table 5.5: Tweet classification performance: Accuracy (Acc), Precision (P) and Recall (R) for the year 2015.

5.4 Results: Tweet Mining

5.4.1 Tweet Classification

The first step for our information extraction task was to classify the tweets as relevant (informative) or non-relevant (non-informative). Because out of all the tweets posted by NYC_DOT only $\sim 14.08\%$ tweets were relevant.

Ground Truth Preparation: The original tweets were not labelled as relevant or non-relevant. So 2 person manually annotate them as relevant and non-relevant. The kappa value for inter annotator agreement between them is 0.76. Next we move on to feature selection task for tweet classification.

Feature Selection for Classification: The relevant tweets contain four key patterns, which helps to correctly classify them. These key features are as follows:

- **Key Words:** For all the tweets, word frequency distribution analysis is done for both relevant and non-relevant tweets separately, after normalizing them (removing stop words followed by applying stemming). Figure 5.5(a)-(b) show clear distinction in

	Acc(%)	P	R	F-Score	ROC
SMO	94.97	0.951	0.950	0.950	0.915
RF	94.94	0.948	0.949	0.948	0.970
BG	**95.24**	**0.951**	**0.952**	**0.951**	**0.958**
NB	94.97	0.951	0.950	0.950	0.972

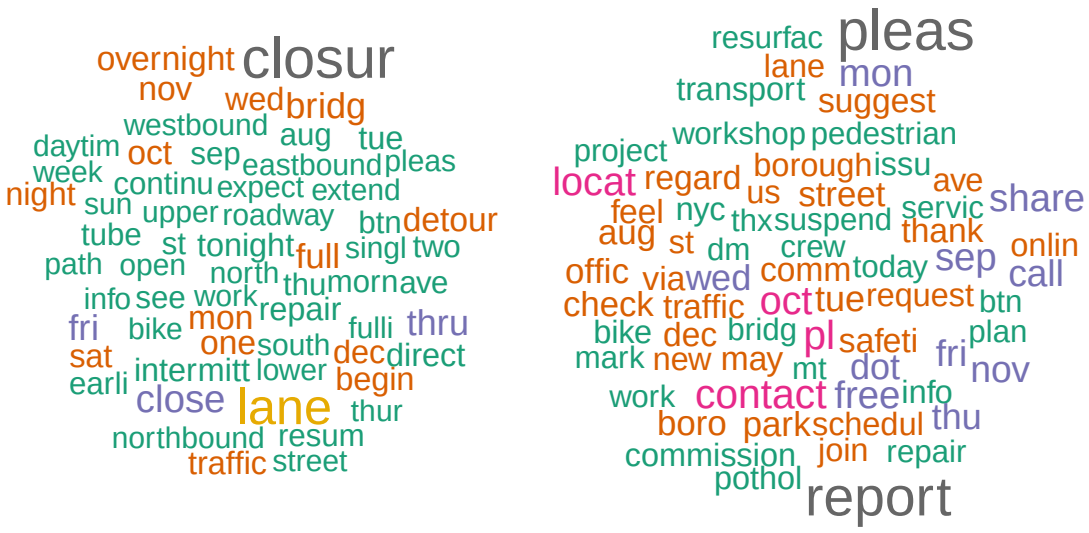
Table 5.6: Tweet classification performance: Accuracy (Acc), Precision (P) and Recall (R) for the year 2016.

word pattern for the relevant and non-relevant post and motivates us for using the top-20 key words as one of our feature.

- **Date:** As it's a notification regarding an upcoming incident, they always mention about the *date* of the event, which makes the tweet distinct from the non-informative one.
- **Time:** These road activities are also specific to a particular *time* say from $10:00$ am to $3:00$ pm. This adds more value to the tweet and helps us in correctly identifying them.
- **Name of the Day:** Finally along with *date* and *time*, the *day* of the week has also been mentioned separately which is also added in the feature list.

We use these four features for training our classifier. The analysis of information gain of the features, in table 5.4, shows *Key Words* (KW), *Time*, *Date* have the highest information gain value and supports our choice of feature list.

Accuracy of classifiers: After generating the data-matrix with above mentioned features, we trained different classifiers using it. The classifiers were implemented using Weka [19]. We report results for the four best classifiers: (1) Sequential Minimization Optimization (SMO) [39], (2) Random Forest (RF) [9], (3) Bagging (BG) [8] and (4) Naive Bayes (NB) classifiers. For SMO, polynomial kernel of degree=3 is used and for Bagging, reduced error pruning tree (REPTree) is used with bag size of 100. Tables 5.5 and 5.6 show the classification accuracy, using $10-$fold cross validation, for tweets of year 2015 and 2016, respectively. Since, we are trying to retrieve all the relevant tweets, therefore our main



(a) (b)

Figure 5.5: Word cloud showing the words used most frequently in the tweet posted by NYC_DOT for (a) relevant (b) non-relevant

Information Type	Success Rate (%)	
	2015	2016
Date-Time Extraction	84.7	88.34
Location Extraction	73.23	72.1

Table 5.7: Information extraction success rate from the relevant tweets for year 2015 and 2016

objective is to build a high recall system. We see for both 2015 and 2016 tweets, Bagging gives the highest average precision $\sim$**0.945**, recall $\sim$**0.945** and accuracy values $\sim$**94.55%**, while other models perform comparably.

5.4.2 Information Extraction Result

After classifying the relevant tweets, next we move on to extract the *date*, *time* and *location* information from each relevant tweet. We use the metric success rate, defined as *% of relevant tweets from which date, time and location names are extracted successfully*, for

measuring the quality of information extraction. Table 5.7 shows that we are able to extract the *date*, *time* information from 84.7% and 88% relevant tweets for year 2015 and 2016 respectively. However in case of location extraction our success rate is around 73.23% and 72.1% for year 2015 and 2016 respectively. The low success rate is due to image based notifications, which we are not processing, or name ambiguity – different shorter name of the same location.

5.5 Predicting Taxi Pickup Hotspots

Through tweet mining, we see that most of the incident locations are near bridges. However these bridges are one of the primary means for interconnection between different cities and islands in New York. So road closures at bridges forces the taxi driver to take a detour and use other bridge to bypass the closure. This changes the regular travel trajectory, resulting in the change of regular passenger pickup points as shown in figure 5.1(b)-(c) in section 5.1. This motivates us to go for second part of this two-step work i.e. learning of taxi hiring locations during such road closure incidents. Because even if all these road closure incidents are planned and have been notified early by the government, the urban commuters might not always get the impact of it.

If a particular road segment is blocked, the taxi driver has to deviate from their regular travel route, resulting in the change of the pickup patterns around the incident locations and thus creating inconveniences for the urban people. We tackle this problem using the following objective:

Given multiple instances of road closure incidents in the same location, can we predict the taxi pickup hotspots for a new incident occurring there?

We pose this as a supervised learning problem where first we identify useful features for each incident location and train our model. Next for a new incident occurring there, we predict the taxi pickup count for all the neighbouring regions around the incident location and identify the hotspot.

Problem Formulation: Assume that at location *l*, *N* road closure events $\{e_1, e_2, \ldots e_N\}$ have occurred at times $\{t_1, t_2, \ldots, t_N\}$ respectively. Given this information we want to predict the taxi pickup *hotspot* around the event location *l* for a new event. To solve this problem, first we consider a grid G of size $k \times k$ in the neighbourhood centring the location. For each grid cell $g \in G$, we select list of features *F* and create a feature vector $v_g = \{f_{g1}, f_{g2}, \ldots, f_{gk}\}$ using transportation data, where f_{gi} denotes the i^{th} feature value for grid cell g. We have the pickup count y_g for grid cell g that denotes the total number of pickups that occurred when the road-closure incident happened at location *l* at time *t*. Next combining all the feature vector we build our feature matrix $\boldsymbol{A} = [v_1, v_2, \ldots, v_N]^T$ and target vector $\boldsymbol{y} = [y_1, y_2, \ldots, y_N]^T$. Finally, we apply different supervised learning technique, Support Vector Regression, Random Forest, etc., implemented in Weka [19], for learning the predictive model to estimate $\boldsymbol{y}$. We discuss about all the feature selection and learning result in details in section 5.6.

5.5.1 Feature Engineering

One of the main challenges is to come up with appropriate features for the regression task. This is especially true since for a given location, we do not have a large number of repeating events. Hence we had to select a small set of important features. Here we provide the list of features used with explanations:

- **LCD:** As already discussed, to predict the pickup hotspot near an incident, we create a grid around the incident location. The coordinates (lat,lon) of these locations (LCD) are included in the feature list.
- **PHN:** We include the pickup history of neighbourhood (PHN) in our feature set as the total pickup counts for the past W days that occurred during the same time period at location *l*.
- **TOD:** To find the correlation of number of pickup count, with different *time of the day* (TOD), we plot the yearly trip count distribution for different hour of the day for the year 2015. Figure 5.6 (a),(b) show that taxi trip count follows a periodic nature

	SVR			LR			RF			BG		
	Cor	**RAE**	**RRSE**	**Cor**	**RAE**	**RRSE**	**Cor**	**RAE**	**RRSE**	**Cor**	**RAE**	**RRSE**
Battery Park	0.98	13.90	22.05	0.98	14.95	22.02	**0.98**	**11.30**	**18.69**	0.98	12.11	19.76
Manhattan Br	0.97	16.87	26.03	0.97	14.41	24.28	**0.97**	**15.31**	**22.67**	0.965	16.82	26.32
Brooklyn Br	0.93	23.03	37.89	0.93	24.26	37.44	**0.97**	**17.06**	**24.83**	0.96	17.97	27.23
Roosevelt BR	0.97	13.90	25.24	0.97	14.41	25.81	**0.97**	**12.18**	**23.13**	0.97	19.12	25.58
QueensBoro Br	**0.98**	**14.05**	**21.84**	0.97	14.31	24.28	0.96	17.46	28.78	0.95	19.40	33.46
WillsBurg Br	**0.93**	**28.31**	**36.78**	0.93	28.62	37.63	0.87	33.68	48.67	0.87	34.03	49.88
WillAve Br	**0.74**	**35.53**	61.80	0.79	44.32	61.33	0.77	42.63	64.66	0.79	41.45	61.44

Table 5.8: Result showing hotspot learning by different prediction model– Support Vector Regression (SVR), Logistic Regression (LR), Random Forest (RF), Bagging (BG) where Correlation Value (Cor) ranges between (0,1) and relative absolute error (RAE) and root relative squared error (RRSE) are in percentage (%).

throughout the day (weekdays and weekends). From midnight, trip count starts decreasing till morning and then it starts rising and hits the peak at 8 am and continues till midnight. Following this observation we include TOD in our feature list.

- **DOW:** Next we focus to find the effect of different *day of the week* (DOW) over trip count. Figure 5.6(c) shows that the total number of trip count per day increases from Monday to Saturday, then drops at Sunday and again rises on Monday and the cycle continues. This motivates us to consider DOW as one of our feature.

- **DFL:** We assume that if any road closure activity happens at a location then as an immediate effect, it disrupts the traffic dynamics of the closer neighbourhood more than the far out region. So *distance from location* (DFL) becomes an useful criterion in this regard and get placed in the feature set.

5.6 Results: Taxi Pickup Hotspots

In this section, we describe results of experiments for automatically predicting the taxi pickup hotspots. Section 5.6.1 describes the input dataset and grid structure. Section 5.6.2 reports accuracy of taxi pickup prediction, while section 5.6.3 discusses distance of predicted hotspots from actual ones.

5.6.1 Experimental Setup

Input Taxi Data: For the transportation data, the New York City (NYC) Taxi data for the year 2015 has been used. This dataset is made public by NYC Taxi and Limousine Commission[7]. They provide the data for all yellow taxi trips made in NYC since 2009. However we have used this data for the year 2015 only. After initial data processing, cleaning and filtering it has $\sim$ 163 million taxi trip data for year 2015. The key fields of the datasets summarized in Table 5.9.

Field	Meaning
pickup date-time	trip start time
dropoff date-time	trip end time
pickup location	GPS coordinate
dropoff location	GPS coordinate

Table 5.9: Fields showing different attributes from New York City taxi trip dataset for learning pickup hotspot

.

Local Grid: Once road closure locations have been identified, we go for a grid based approach for learning taxi pickup hotspot following the idea of Tejaswin et. al. [52]. A key difference is that they build the grid globally over entire city for classifying each grid, whether any events, accidents will occur or not. Instead, we use the idea of localized grid around the road closure locations and estimate the pickup count in these grids during the event. For each location we take a square region of size 2×2 sq. miles centring the location. Next we break the region in size of 0.2×0.2 sq. miles. Now we move on to the hotspot learning task as discussed in section 5.5. For each grid location we build the feature matrix $\boldsymbol{A}$ and the target vector $\boldsymbol{y}$ and learn our model. After, estimating the pickup count for each cell, we apply minimum pickup count threshold for labelling each cell as hotspot or not. In this work, our minimum pickup count threshold is set to 3.

[7] http://www.nyc.gov/html/tlc/html/about/trip record data.shtml

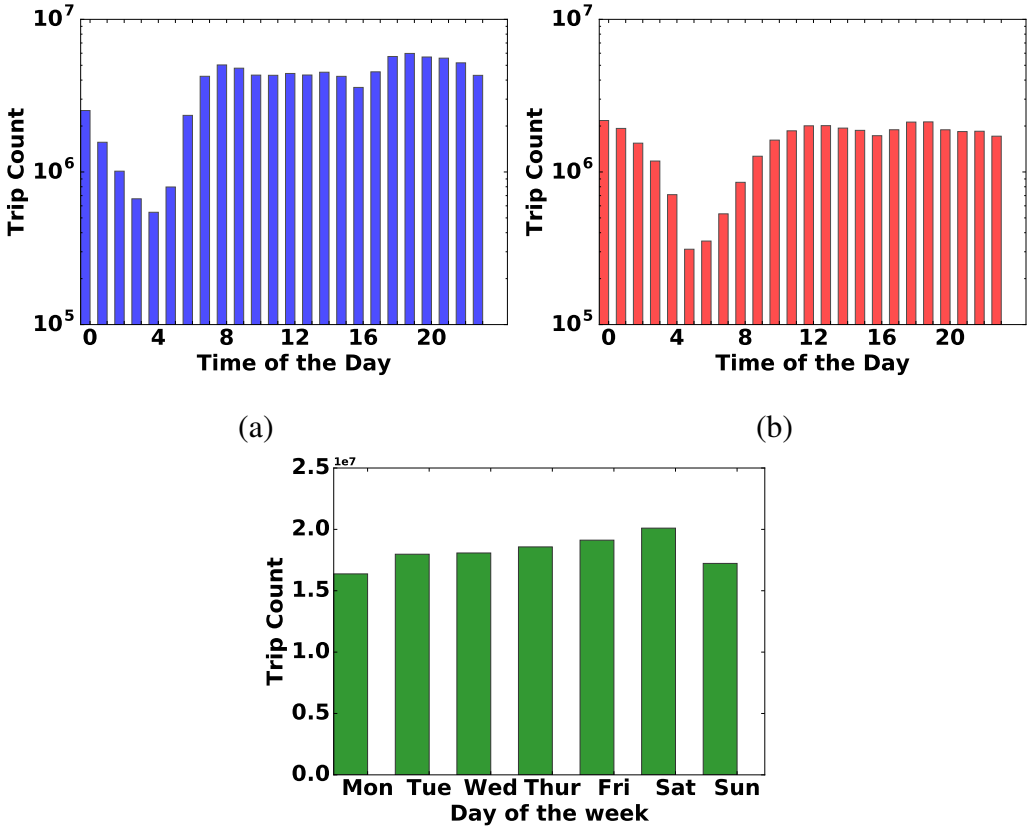


Figure 5.6: Distribution of trip count at different hours of the day for (a) Weekday (b) Weekend (c) Weekly.

5.6.2 Predicting Taxi Pickup

After we select the features, next we build our feature vector v_g as a collection of related features as $v_g = \{LCD_g, PHN_g, TOD_g, DOW_g, DFL_g\}$ for each neighbouring region. Next we build our feature matrix $\boldsymbol{A}$ and target vector $\boldsymbol{y}$ as discussed in section 5.5 and apply different learning algorithm for predicting the pickup count in the neighbouring locations. To measure the accuracy of our learning system, we calculate the Relative Absolute Error (RAE) and Root Relative Squared Error (RRSE). If for i^{th} observation, the predicted value is $\hat{y}_i$ and the observed value is y_i then RAE and RRSE are calculated as follows:

$$RAE = \frac{\sum_{i=i}^{N} |\hat{y}_i - y_i|}{\sum_{i=i}^{N} |\bar{y}_i - y_i|} * 100 \tag{5.1}$$

$$RRSE = \sqrt{\frac{\sum_{i=i}^{N}(\hat{y}_i - y_i)^2}{\sum_{i=i}^{N}(\bar{y}_i - y_i)^2}} * 100 \tag{5.2}$$

Finally RAE and RRSE are multiplied with 100 to set the scale in $0 - 100$. Here, the values are normalized by how much y differs from it's mean value.

For our learning methodology we have used Weka [19] and have applied different supervised learning technique. However we report only the top four technique which are giving the best result namely (SVR) [48], [50], Random Forest [9], Bagging [8] and Linear Regression. Here, in SVR, we have used polynomial kernel with degree 3. For Bagging, we use reduced error pruning tree (REPTree) with bag size of 100. Also due to the smaller number of data size, we have used $10-$fold cross validation technique to measure the performance of different models. We have applied this technique on all the seven major incident locations for predicting the taxi pickup count in the neighbouring regions. Table 5.8 shows the performance of the learning technique on these locations. We see that Random Forest (RF) and SVR are giving the most promising result. The table shows that Random Forest (RF) gives good result for Battery Park, Manhattan Bridge, Brooklyn Bridge and Roosevelt Bridge compared to the other three model. Here the average correlation score between the observed and the predicted value is 0.975%. Whereas the average RAE is $\sim 13.96\%$, RRSE is $\sim 22.33\%$ for these four locations. On the other hand, for Queens-Boro Bridge, SVR is giving comparatively better result compared to Random Forest. Here the average correlation between the observed and the predicted is 0.98, RAE is 14% and RRSE is 21.84%. But for Williamsburg Bridge and Wills Avenue Bridge the result is not that good where average correlation is only 0.835. The reason for such weak performance of the model is due to the smaller data size as comparatively fewer number of road closures incidents occurs in these places.

5.6.3 Distance of Predicted Hotspots

Once we estimate the taxi pickup count during a new road closure event at a location, we apply minimum pickup threshold, *Th* = 3, and labels them as hotspot or not. Thus we obtain our predicted taxi pickup hotspots. Next we measure the similarity between the predicted taxi pickup hotspots with the ground truth hotspots during an event. The second column of

Location	Similarity (%)	Avg Distance Error (mile)
Battery Park	40	0.042
Brooklyn Bridge	79	0.013
Williamsburg Bridge	93	0.004
WillAve Bridge	100	0.000
QueensBoro Bridge	87	0.011
Roosevelt Bridge	64	0.001

Table 5.10: Comparison of Predicted Taxi Pickup Hotspot against the original hotspots.

table 5.10 shows the location similarity of the predicted event specific taxi pickup hotspot (ESTPH) with the original ESTPH. We see that WillAve Bridge and Williamsburg Bridge get the highest similarity, 100% and 93% respectively. The third column shows, the average distance error (in mile) between the predicted ESTPH and the original ESTPH. We see, that the average distance error is only ~ 0.011 mile. This shows, how close our predicted ESTPHs are to the original ESTPHs, assuring the success of the hotspot prediction. Thus we see that, our model is predicting the event specific taxi pickup hotspots, which are very close to the actual one.

5.7 Conclusion

The contribution of this work is manifold. Here, we have been able to capture the taxi pickup hotspot relocation due to road closure events, by analysing the event notification in the online social network and taxi transportation data. Perhaps we are the first to to understand the pickup hotspot relocation using social network and past road closure trends from transportation network. A novel two-step process is proposed for predicting the taxi pickup hotspot from the traffic notifications posted on twitter. In the first step the relevant tweets are classified with average 94.55% accuracy, ~ 0.945 precision and ~ 0.945 recall (Table 5.5 and 5.6). In the second step, we proceed for predicting the taxi hiring hotspots leveraging the inferred knowledge base and past pickup trend during road closure. Our supervised learning technique achieves an average $\sim 13.96\%$ root absolute error (RAE) and

$\sim 22.33\%$ root relative squared error (RRSE) using Random Forest technique for locations with higher number of road blocking incidents. The predicted hotspots are also within an average radius of only 0.011 mile from the actual hotspots. This shows, how close the predicted hotspots are to the original hotspots. The prediction has the potential of suggesting the experienced as well as new comers in the city, about the possible taxi pickup hotspots if any new road closure events occur at a place, based upon their past experience.

Chapter 6

Conclusion

In this chapter, we summarize the main contributions of the thesis and outline our achievements in comparison to the objectives set up in the introductory chapter. Finally, we wrap up by pointing out some of the possible future directions of research that have been opened up by this thesis.

6.1 Summary of the contributions

In this thesis, our contributions are three-fold: (a) estimation of link travel time and its variance from large scale end-point trip data, which provide a measure of path reliability in a transportation network; (b) covariance estimation of link travel durations from covariance of path travel durations, which allows for an understanding of how traffic in the links are correlated with each other; (c) prediction of taxi pickup hotspots during road-link closure incidents by mining Twitter and New York City taxi data, which helps in deciphering the dynamics of taxi pickup hotspots during man-made urban events.

Estimation of link travel time and its variance

We investigate the problem of estimating link travel time and variance in link travel time for any two given points from incomplete end-to-end data. Next, we build a route

recommendation system using the estimated link attribute.

- Recently, New York city Taxi and Limousine Commission[1] has freely shared vast amounts of data for about all taxi trips made in New York city, during 2014. The dataset provides the starting and ending locations and timestamps along with total distance travelled, fare, etc., for each trip; but does not include routes taken by the taxi. Our framework maps 65.6 million taxi trips, recorded in Manhattan during 2014, to unambiguous paths, within 0.1 mile error threshold.

- The mean travel time along with its variance are accordingly estimated. Results from experiments with link travel time prediction algorithms show that mean absolute percentage error (MAPE) for predicted link travel times is better by at least 20% than existing methods. Also, the root mean square error for path variances estimation is within at most 1.5 minute on an average, i.e., for an observed path variance of 5 minutes, our estimated variance lies within $3.5 - 6.5$ minutes, confirming the effectiveness of our variance estimation method.

- We also demonstrate that the estimated link travel time and variance can be effectively used to recommend paths, thereby introducing the concept of certainty based route recommendation. The performance of our route recommendation system is also comparable to commercial systems. Note that the performance of systems like Google Maps and Waze is based on real-time data from individual users, while we work on historical (2014) data. Interestingly, we find that the routes followed by our certainty based scheme are different from routes suggested by Google Maps, and are also less prone to on-road uncertainties. We are perhaps being able to suggest alternate paths due to the scalability of our system, which enables us to capture a holistic view of the city $-$ however, this claim requires further investigation.

[1] http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

Estimation of link covariance

We move on to estimate link covariance in travel time for understanding link-link correlations, and their types – positive or negative. The main motivation is that during an event in the city, it is crucial to assess individual links to understand how the traffic in one link is affecting another, which in turn influences the overall traffic dynamics.

- Here we estimate covariance between link travel durations given the covariance values for path travel durations. We call this our *Dependence Link Variance* (DLV) model where we assume that traffic in one link has correlation with traffic in another link. Results from experiments show that the DLV method is better by at least 12% than the *Independent Link Variance* (ILV) method, which is based on the assumption that there is no link correlation. Result shows that the predicted links are within an average error radius of ~ 0.2 mile from the original traffic prone links.
- Our sparse inverse link covariance estimation captures the most correlated links, which are in close vicinity to the congested region. Ours is, to the best of our knowledge, the first work to identify long distance link correlations. We observe that $\sim 30\%$ positively correlated links are more than 1.0 mile apart.

Prediction of taxi pickup hotspots during road closure incidents

Finally, we study how traffic behaviour changes around a closed link, using the help of transportation data and the traffic feed from online social networking sites. Perhaps we are the first to study relocations of taxi pickup hotspots, as an impact of road link disruptions.

- A novel two-step process is proposed for predicting relocation of taxi pickup hotspots using traffic notifications posted on Twitter. In the first step, the relevant tweets are classified with an average of 94.55% accuracy, ~ 0.945 precision, and ~ 0.945 recall.
- In the second step, we predict taxi hiring hotspots leveraging upon the inferred knowledge base and past pickup trends during road closure. Our supervised learning technique achieves an average of $\sim 13.96\%$ root absolute error (RAE) and $\sim 22.33\%$ root relative squared error (RRSE) using Random Forest technique for locations with higher number of road blocking incidents.

- The predicted hotspots are within an average radius of only 0.011 mile from the actual hotspots. Such an accurate prediction mechanism has the potential of suggesting possible pickup hotspots to experienced commuters as well as new-comers in the city.

6.2 Directions of future work

In this final section, we outline a few out of the many possible directions of future research that have been opened up by this thesis. One direction could be to investigate the link correlations during various types of global events such as earthquakes, snowfall, hurricanes, etc., in a city wide scale. This will help in understanding the resilience of the transportation network of the entire city during extreme conditions, which would enable city authorities to effectively plan evacuation, sending of emergency logistics, medical aid, etc., in a more efficient way.

Another line of research could be to investigate passengers' mobility patterns and drivers' travel patterns during road-link disruptions. Since such events increase congestion in the neighbouring road network, better understanding of such dynamics enable micro-level traffic management.

Bibliography

[1] The bright side of sitting in traffic: Crowdsourcing road congestion data. https://googleblog.blogspot.in/2009/08/bright-side-of-sitting-in-traffic.html, 2009.

[2] A. Agovic, A. Banerjee, A. R. Ganguly, and V. Protopopescu. Anomaly detection in transportation corridors using manifold embedding. *Knowledge Discovery from Sensor Data*, pages 81–105, 2008.

[3] M. Asghari, T. Emrich, U. Demiryurek, and C. Shahabi. Probabilistic estimation of link travel times in dynamic road networks. In *SIGSPATIAL*, pages 47:1–47:10, 2015.

[4] H. Bast, S. Funke, P. Sanders, and D. Schultes. Fast routing in road networks with transit nodes. *Science*, 316(5824):566–566, 2007.

[5] H. Becker, D. Iter, M. Naaman, and L. Gravano. Identifying content for planned events across social media sites. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 533–542. ACM, 2012.

[6] E. Benson, A. Haghighi, and R. Barzilay. Event discovery in social media feeds. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 389–398. Association for Computational Linguistics, 2011.

[7] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[8] L. Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[9] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[10] S. Chawla, Y. Zheng, and J. Hu. Inferring the root cause in road traffic anomalies. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 141–150. IEEE, 2012.

[11] B. Y. Chen, W. H. Lam, A. Sumalee, and Z.-l. Li. Reliable shortest path finding in stochastic networks with spatial correlated link travel times. *International Journal of Geographical Information Science*, 26(2):365–386, 2012.

[12] M. Chen and S. Chien. Dynamic freeway travel-time prediction with probe vehicle data: Link based versus path based. *Transportation Research Record: Journal of the Transportation Research Board*, (1768):157–161, 2001.

[13] Z. Chen, H. T. Shen, and X. Zhou. Discovering popular routes from trajectories. In *Data Engineering (ICDE)*, pages 900–911. IEEE, 2011.

[14] T. H. Cormen. *Introduction to algorithms*. MIT press, 2009.

[15] E. M. Daly, F. Lecue, and V. Bicer. Westland row why so slow?: fusing social media and linked data sources for understanding real-time traffic conditions. In *Proceedings of the 2013 international conference on Intelligent user interfaces*, pages 203–212. ACM, 2013.

[16] E. W. Dijkstra. A note on two problems in connexion with graphs:(numerische mathematik, _1 (1959), p 269-271). 1959.

[17] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[18] R. Geisberger, P. Sanders, D. Schultes, and C. Vetter. Exact routing in large road networks using contraction hierarchies. *Transportation Science*, 46(3):388–404, 2012.

[19] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, Nov. 2009.

[20] E. Jenelius and H. N. Koutsopoulos. Travel time estimation for urban road networks using low frequency probe vehicle data. *Transportation Research Part B: Methodological*, 53:64–81, 2013.

[21] B. Kulis, A. C. Surendran, and J. C. Platt. Fast low-rank semidefinite programming for embedding and clustering. In *AISTATS*, pages 235–242, 2007.

[22] T. Kurashima, T. Iwata, G. Irie, and K. Fujimura. Travel route recommendation using geotags in photo sharing sites. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 579–588. ACM, 2010.

[23] T. Lancewicki and M. Aladjem. Multi-target shrinkage estimation for covariance matrices. *IEEE Transactions on Signal Processing*, 62(24):6380–6390, 2014.

[24] F. Lécué, R. Tucker, V. Bicer, P. Tommasi, S. Tallevi-Diotallevi, and M. Sbodio. Predicting severity of road traffic congestion using semantic web technologies. In *European Semantic Web Conference*, pages 611–627. Springer, 2014.

[25] O. Ledoit and M. Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of multivariate analysis*, 88(2):365–411, 2004.

[26] Y. Li. Short-term prediction of motorway travel time using anpr and loop data. *Journal of Forecasting*, 27(6):507–517, 2008.

[27] R. J. Little and D. B. Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2014.

[28] L. Liu, J. Xu, S. S. Liao, and H. Chen. A real-time personalized route recommendation system for self-drive tourists based on vehicle to vehicle communication. *Expert Systems with Applications*, 41(7):3409–3417, 2014.

[29] W. Luo, H. Tan, L. Chen, and L. M. Ni. Finding time period-based most frequent path in big trajectory data. In *SIGMOD 2013*, pages 713–724.

[30] Z. Ma, H. N. Koutsopoulos, L. Ferreira, and M. Mesbah. Estimation of trip travel time distribution using a generalized markov chain approach. *Transportation Research Part C: Emerging Technologies*, 74:1–21, 2017.

[31] H. Mo, X. Hao, and D. Wen. Linguistic dynamic analysis of traffic flow based on social mediaa case study.

[32] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas. Predicting taxi–passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1393–1402, 2013.

[33] Y. M. Nie and X. Wu. Reliable a priori shortest path problem with limited spatial and temporal dependencies. In *Transportation and traffic theory 2009: golden jubilee*, pages 169–195. Springer, 2009.

[34] E. V. Nikolova. *Strategic algorithms*. PhD thesis, Massachusetts Institute of Technology, 2009.

[35] B. Pan, U. Demiryurek, and C. Shahabi. Utilizing real-world transportation data for accurate traffic prediction. In *2012 IEEE 12th International Conference on Data Mining*, pages 595–604. IEEE, 2012.

[36] B. Pan, Y. Zheng, D. Wilkie, and C. Shahabi. Crowd sensing of traffic anomalies based on human mobility and social media. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 344–353. ACM, 2013.

[37] Y. Pan, L. Sun, and M. Ge. Finding reliable shortest path in stochastic time-dependent network. *Procedia-Social and Behavioral Sciences*, 96:451–460, 2013.

[38] A. Pathak, B. K. Patra, A. Chakraborty, and A. Agarwal. A city traffic dashboard using social network data. In *Proceedings of the 2Nd IKDD Conference on Data Sciences*, CODS-IKDD '15, pages 8:1–8:4, New York, NY, USA, 2015. ACM.

[39] J. Platt et al. Sequential minimal optimization: A fast algorithm for training support vector machines. 1998.

[40] Z. S. Qian, J. Li, X. Li, M. Zhang, and H. Wang. Modeling heterogeneous traffic flow: A pragmatic approach. *Transportation Research Part B: Methodological*, 99:183–204, 2017.

[41] M. Rahmani, E. Jenelius, and H. N. Koutsopoulos. Non-parametric estimation of route travel time distributions from low-frequency floating car data. *Transportation Research Part C: Emerging Technologies*, 58:343–362, 2015.

[42] T. H. Rashidi, A. Abbasi, M. Maghrebi, S. Hasan, and T. S. Waller. Exploring the capacity of social media data for modelling travel behaviour: Opportunities and challenges. *Transportation Research Part C: Emerging Technologies*, 75:197–211, 2017.

[43] A. Ritter, O. Etzioni, S. Clark, et al. Open domain event extraction from twitter. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1104–1112. ACM, 2012.

[44] I. Sanaullah. *Real-time estimation of travel time using low frequency GPS data from moving sensors*. PhD thesis, © Irum Sanaullah, 2013.

[45] J. Schäfer, K. Strimmer, et al. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical applications in genetics and molecular biology*, 4(1):32, 2005.

[46] D. L. Schrank and T. J. Lomax. *2009 urban mobility report*. Texas Transportation Institute, Texas A & M University, 2009.

[47] C. Shen, A. Welsh, and L. Wang. Psdboost: Matrix-generation linear programming for positive semidefinite matrices learning. In *Advances in Neural Information Processing Systems*, pages 1473–1480, 2009.

[48] S. K. Shevade, S. S. Keerthi, C. Bhattacharyya, and K. R. K. Murthy. Improvements to the smo algorithm for svm regression. *IEEE transactions on neural networks*, 11(5):1188–1193, 2000.

[49] V. P. Sisiopiku and N. M. Rouphail. Toward the use of detector output for arterial link travel time estimation: a literature review. *Transportation Research Record*, (1457), 1994.

[50] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.

[51] P. B. Stark and R. L. Parker. Bounded-variable least-squares: an algorithm and applications. *Computational Statistics*, 10:129–129, 1995.

[52] P. Tejaswin, R. Kumar, and S. Gupta. Tweeting traffic: Analyzing twitter for generating real-time city traffic insights and predictions. In *Proceedings of the 2nd IKDD Conference on Data Sciences*, page 9. ACM, 2015.

[53] J. L. Toole, S. Colak, B. Sturt, L. P. Alexander, A. Evsukoff, and M. C. González. The path most traveled: Travel demand estimation using big data resources. *Transportation Research Part C: Emerging Technologies*, 58:162–177, 2015.

[54] A. Touloumis. Nonparametric stein-type shrinkage covariance matrix estimators in high-dimensional settings. *Computational Statistics & Data Analysis*, 83:251–261, 2015.

[55] M. Treiber and A. Kesting. Travel time estimation. In *Traffic Flow Dynamics*, pages 367–377. Springer, 2013.

[56] H. Wang, Z. Li, Y.-H. Kuo, and D. Kifer. A simple baseline for travel time estimation using large-scale trip data. *arXiv preprint arXiv:1512.08580*, 2015.

[57] Y. Wang. Socializing multimodal sensors for information fusion. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 653–656. ACM, 2015.

[58] Y. Wang, Y. Zheng, and Y. Xue. Travel time estimation of a path using sparse trajectories. In *SIGKDD 2014*, pages 25–34.

[59] L.-Y. Wei, Y. Zheng, and W.-C. Peng. Constructing popular routes from uncertain trajectories. In *SIGKDD 2012*, pages 195–203.

[60] F. Wu, H. Wang, and Z. Li. Interpreting traffic dynamics using ubiquitous urban data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 69. ACM, 2016.

[61] J. Y. Yen. Finding the k shortest loopless paths in a network. *management Science*, 17(11):712–716, 1971.

[62] X. Zhan, S. Hasan, S. V. Ukkusuri, and C. Kamga. Urban link travel time estimation using large-scale taxi data with partial information. *Transportation Research Part C: Emerging Technologies*, 33:37–49, 2013.

[63] X. Zhan, S. V. Ukkusuri, and C. Yang. A bayesian mixture model for short-term average link travel time estimation using large-scale limited information trip-based data. *Automation in Construction*, 2015.

[64] F. Zheng and H. Van Zuylen. Urban link travel time estimation based on sparse probe vehicle data. *Transportation Research Part C: Emerging Technologies*, 31:145–157, 2013.

[65] Y. Zhou, W. Wang, D. He, and Z. Wang. A fewest-turn-and-shortest path algorithm based on breadth-first search. *Geo-spatial Information Science*, 17(4):201–207, 2014.

Appendix A

Publications from the Thesis

Publications from the work presented in the thesis: The publications are listed in chronological order with comments in parenthesis.

1. Sankarshan Mridha, Sayan Ghosh, Robin Singh, Sourangshu Bhattacharya and Niloy Ganguly. "Mining Twitter and Taxi Data for Predicting Taxi Pickup Hotspots", *accepted in the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), Sydney, Australia 2017.* (This study is presented in Chapter 5)