

Faculty of Engineering and Information Technology  
University of Technology Sydney

# **Deep Learning for Traffic Time Series Data Analysis**

A thesis submitted in partial fulfillment of  
the requirements for the degree of  
**Doctor of Philosophy**

by

Xiaocai Zhang

November 2020



## CERTIFICATE OF AUTHORSHIP/ORIGINALITY

I, Xiaocai ZHANG declare that this thesis, is submitted in fulfilment of the requirements for award of Doctor of Philosophy, in the Advanced Analytics Institute, Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is support by the Australian Government Research Training Program.

Production Note:  
SIGNATURE OF CANDIDATE: Signature removed prior to publication.  
[Xiaocai Zhang]

DATE: 26<sup>th</sup> Nov, 2020



# Acknowledgments

First of all, I would like to express my sincere thankfulness to my principal supervisor Prof. Jinyan Li for his continuous support, assistance and advice during my Ph.D. studies. I would further like to thank other members in Prof. Jinyan's research team and the Advanced Analytics Institute for their kind helps on my research and daily life, they are Dr. Hui Peng, Dr. Yi Zheng, Dr. Chaowang Lan, Dr. Yuansheng Liu, Dr. Zhixun Zhao, Xuan Zhang, Tao Tang, Tian Lan and Dr. Sunny Verma. Thanks for all the memorized scenes they brought to my life.

I am also grateful to my co-supervisor Dr. Wei Liu, for his kindly helps during my Ph.D. career. Furthermore, I would like to extend my sincere gratitude to Prof. Michael Blumenstein and A/Prof. Yang Wang for their helpful suggestions on my research work.

I sincerely acknowledge the organizations that provide financial supports for this research, including the China Scholarship Council and the University of Technology Sydney.

I would like to thank all the anonymous peer reviewers for their insightful comments and suggestions to significantly improve the quality of this work.

Lastly, I would like to express my deepest gratitude to my family members and friends, who were always encouraging me when I was in difficult times.

Xiaocai Zhang @ Sydney  
Nov 2020



# Contents

Certificate of Authorship/Originality . . . . .	i
Acknowledgment . . . . .	iii
List of Figures . . . . .	ix
List of Tables . . . . .	xi
List of Publications . . . . .	xiii
Nomenclature . . . . .	xv
Abstract . . . . .	xvii
<b>Chapter 1 Introduction . . . . .</b>	<b>1</b>
1.1 Background . . . . .	1
1.1.1 Traffic Time Series Data . . . . .	2
1.1.2 Deep Learning . . . . .	4
1.1.3 Applications of Traffic Time Series Data Analysis . . . . .	5
1.2 Research Motivations . . . . .	7
1.2.1 Taxi Destination Prediction . . . . .	7
1.2.2 Anomalous Traffic Patterns Detection . . . . .	8
1.2.3 Traffic Flow Prediction . . . . .	9
1.3 Research Objectives and Contributions . . . . .	10
1.4 Thesis Structure . . . . .	12
<b>Chapter 2 Related Work and Literature Review . . . . .</b>	<b>14</b>
2.1 Taxi Trajectory Modelling . . . . .	14
2.1.1 Recommendation Systems . . . . .	14
2.1.2 Location Prediction . . . . .	15

2.2	Anomalous Patterns Detection . . . . .	17
2.2.1	General Anomaly Detection . . . . .	17
2.2.2	Road Traffic Anomaly Detection . . . . .	19
2.3	Traffic Flow Prediction . . . . .	20
2.3.1	Parametric Methods . . . . .	21
2.3.2	Non-parametric Methods . . . . .	21
2.4	Summary . . . . .	23
 <b>Chapter 3 Taxi Destinations Prediction from Trajectories Using a Novel Data Embedding Method and Ensemble Learning . . . . . 24</b>		
3.1	Methods . . . . .	25
3.1.1	Circular Fuzzy Embedding (CFE) . . . . .	25
3.1.2	Ensemble Learning Model (ELM) . . . . .	30
3.1.3	Input and Output Layers . . . . .	32
3.2	Experiments and Results . . . . .	34
3.2.1	Datasets and Evaluation Metrics . . . . .	34
3.2.2	Experimental Settings . . . . .	35
3.2.3	Evaluation on the Constructed Classifier . . . . .	39
3.2.4	Comparison with Baseline Methods . . . . .	40
3.2.5	Evaluation on the Proposed CFE . . . . .	41
3.2.6	Case Study . . . . .	43
3.3	Summary . . . . .	43
 <b>Chapter 4 Offline and Online Detection of Anomalous Patterns from Bus Trajectories for Traffic Insight Analysis . . . . . 45</b>		
4.1	Methods . . . . .	47
4.1.1	Preliminaries . . . . .	47
4.1.2	Feature Extraction and Trajectory Visualization Using Deep Learning . . . . .	48
4.1.3	Offline Anomalous Traffic Patterns Detection . . . . .	50



4.1.4	Insight Analysis Using Anomalous Patterns . . . . .	54
4.1.5	Online Detection of Anomalous Traffic Patterns . . . . .	55
4.2	Experiments and Results . . . . .	56
4.2.1	Datasets and Evaluation metrics . . . . .	57
4.2.2	Parameters . . . . .	58
4.2.3	Offline Detection Results about Anomalous Patterns . . . . .	59
4.2.4	Results about Feature Visualization and Anomaly Insight Analysis . . . . .	61
4.2.5	Online Detection Results about Anomalous Patterns . . . . .	65
4.2.6	Comparison with Baseline Methods . . . . .	68
4.3	Summary . . . . .	70
<b>Chapter 5 Differential Evolution based LSTM Recurrent Neural Network for Traffic Flow Prediction . . . . .</b>		<b>72</b>
5.1	Methods . . . . .	74
5.1.1	Differential Evolution . . . . .	75
5.1.2	Differential Evolution based LSTM under Parallel Computing . . . . .	76
5.1.3	PDE-LSTM: An Illustrative Example . . . . .	78
5.2	Experiments and Results . . . . .	81
5.2.1	Datasets and Evaluation Metrics . . . . .	81
5.2.2	Hyperparameters . . . . .	83
5.2.3	Parameter Settings . . . . .	84
5.2.4	Prediction Performances Comparison . . . . .	85
5.2.5	Residual and Correlation Analyses . . . . .	87
5.2.6	Runtime Comparison . . . . .	89
5.3	Summary . . . . .	90
<b>Chapter 6 Conclusion and Future Work . . . . .</b>		<b>91</b>
6.1	Conclusion . . . . .	91
6.2	Future Work . . . . .	93

Chapter A Appendix: Supplementary Files . . . . .	95
Chapter B Appendix: List of Abbreviations . . . . .	96
Bibliography . . . . .	101

# List of Figures

1.1	Thesis structure. . . . .	13
3.1	Circle constructed for feature of hour of day. . . . .	26
3.2	(a) Circle constructed for feature of day type. (b) Circle constructed for feature of week of year. . . . .	27
3.3	Linear combined membership function of hour of day feature. .	28
3.4	Workflow chart of ELM. . . . .	32
3.5	Architecture of DBN model. . . . .	33
3.6	Determination of $\lambda$ with validation sets. (a) on the Porto dataset. (b) on the Chengdu dataset. . . . .	37
3.7	Network error with epoch in DBN. (a) on the Porto dataset. (b) on the Chengdu dataset. . . . .	39
3.8	Accuracy of kNN classifier for estimating current segment. . .	40
3.9	Comparison of ELM and other models with a case study (on Porto dataset). . . . .	42
4.1	The workflow of offline and online detection of anomalous traffic patterns and anomaly insight analysis. . . . .	46
4.2	Performance comparisons under different settings of parameter $\delta$ . .	60
4.3	Examples of color trajectories generated by DSAE, PCA, RP or SSAE. Each method generates similar visualization patterns on all of these datasets. The CT visualizations generated by our proposed DSAE model are the smoothest. . . . .	63

4.4	Insight analyses for anomaly #1 in Bus Route 66 and anomaly #4 in Bus Route 50. (i) CTM of the anomalous trajectory. (ii) CT of the anomalous trajectory. (iii) CT of a non-anomalous trajectory. (iv) CT of another non-anomalous trajectory. . . .	64
4.5	Comparison of concentration performance on the training sets.	65
4.6	An illustrative example of online traffic anomaly detection process in Bus Route 66. . . . .	67
5.1	The workflow of PDE-LSTM in traffic flow prediction. . . . .	74
5.2	Average traffic flow of the M50-N, M1-N and I280-S roads during a typical week. . . . .	82
5.3	Comparison between the predicted flow and the groundtruth flow on the 15-min prediction task. The predicted traffic flow and trends (denoted by red dash line) by our PDE-LSTM closely match with the groundtruth traffic flow (denoted by blue solid line) for all of the test sets. . . . .	86
5.4	Residual error analysis for the 15-min prediction by PDE-LSTM model. (a)(b)(c) plot of residual error. (d)(e)(f) histogram of residual error . . . . .	88

# List of Tables

1.1	Sensor technologies of traffic time series data collection . . . .	3
3.1	Sizes of training, test and validation sets . . . . .	36
3.2	Effect of different structures on validation set (on the Porto dataset) . . . . .	38
3.3	Effect of different structures on validation set (on the Chengdu dataset) . . . . .	38
3.4	Time costs by the our proposed model . . . . .	42
3.5	Distances between the real and predicted destinations with different models (unit: km) . . . . .	43
4.1	Detected anomalies for each bus route . . . . .	62
4.2	Performance of proposed online anomaly detection method (ON-ATPD) . . . . .	66
4.3	Performance comparison on the test sets with the baseline methods . . . . .	69
5.1	Population with the step of initialization . . . . .	79
5.2	Population with the step of mutation . . . . .	79
5.3	Population with the step of crossover . . . . .	79
5.4	Fitness values of initialized population (target vectors) . . . .	80
5.5	Fitness values of crossovered population (trail vectors) . . . .	80
5.6	Offspring population with the step of selection . . . . .	80
5.7	Dataset description . . . . .	83

*List of Tables*

---

5.8	Lower and upper bounds of each hyperparameter . . . . .	85
5.9	Computational time cost comparison . . . . .	90

# List of Publications

## Journal Papers :

- J-1: **Zhang X.**, Zhao Z., Zheng Y. & Li J. (2020), ‘Prediction of taxi destination using a novel data embedding method and ensemble learning’, *IEEE Transactions on Intelligent Transportation Systems* 21(1), 68-78.
- J-2: **Zhang X.**, Zheng Y., Zhao Z., Liu Y., Blumenstein M. & Li J. (2020), ‘Anomalous patterns detection from bus trajectories for traffic insight analysis’, submitted to *Knowledge-Based Systems*. (Under review).
- J-3: **Zhang X.**, Zhao Z. & Li J. (2020), ‘Differential evolution based lstm recurrent neural network: a deep learning approach for traffic flow prediction’, submitted to *IEEE Transactions on Intelligent Transportation Systems*. (Under revision).
- J-4: Zheng Y., Peng H., **Zhang X.**, Zhao Z., Gao X. & Li J. (2019), ‘Old drug repositioning and new drug discovery through similarity learning from drug-target joint feature spaces’, *BMC Bioinformatics*, 20(S23):605.
- J-5: Zhao Z., Peng H., **Zhang X.**, Zheng Y., Chen F., Fang L. & Li J. (2019), ‘Identification of lung cancer gene markers through kernel maximum mean discrepancy and information entropy’, *BMC Medical Genomics*, 12(S8):183.

- J-6: Zheng Y., Peng H., **Zhang X.**, Zhao Z., Yin J. & Li J. (2018), ‘Predicting adverse drug reactions of combined medication from heterogeneous pharmacologic databases’, *BMC Bioinformatics*, 19(S19):517.
- J-7: Zhao Z., **Zhang X.**, Chen F., Fang L. & Li J. (2020), ‘Accurate prediction of DNA N4-methylcytosine sites via boost-learning various types of sequence features’, *BMC Genomics*, 21(1), 1-11.

## Conference Papers :

- C-1: **Zhang X.**, Zhang X., Verma S., Liu Y., Blumenstein M. & Li J. (2019), Detection of anomalous traffic patterns and insight analysis from bus trajectory data, *in* ‘Pacific Rim International Conference on Artificial Intelligence’. Springer, pp. 307-321.
- C-2: **Zhang X.**, Liu Y., Zheng Y., Zhao Z., Li J. & Liu Y. (2018), Distinction between ships and icebergs in sar images using ensemble loss trained convolutional neural networks, *in* ‘Australasian Joint Conference on Artificial Intelligence’. Springer, pp. 216-323.
- C-3: Zheng Y., Peng H., **Zhang X.**, Gao X. & Li J. (2018), Predicting drug targets from heterogeneous spaces using anchor graph hashing and ensemble learning, *in* ‘International Joint Conference on Neural Networks’. IEEE, pp. 1-7.



# Nomenclature

$\mathbf{A}$	Matrix
$\mathbf{a}$	Vector
$a$	Scalar
$\mathbf{W}$	Weight matrix
$\mathbf{b}$	Bias vector
$\mathbf{X}$	Dataset input
$\mathbf{Y}$	Dataset output
$\mathbf{x}_i$	Input data vector
$\mathbf{y}_i$	Output data vector
$x_{ij}$	Input data point
$y_{ij}$	Output data point
$\hat{\mathbf{y}}$	Groundtruth vector
$\hat{y}_i$	Groundtruth data point
$e_i$	Residual error
$(.)^T$	The transpose operation
$\mathbb{R}$	The field of real numbers



# Abstract

Time series data in traffic has been playing an important role in intelligent transportation systems (ITS) research and applications. However, because of the sparse, imbalanced, stochastic and highly non-linear natures of traffic time series data, typical parametric methods or machine learning methods are unable to learn the representations of data well. This work aims to develop deep learning methods to gain novel and valuable knowledge from traffic time series analysis for ITS. Specifically, deep learning-based methods are developed for three topics, namely taxi destination prediction, anomalous traffic patterns detection, and urban traffic flow prediction.

The first method is to predict taxi destination using trajectory data. Accurate and timely destination prediction of taxis is of great importance for location-based service applications. Over the last few decades, popularization of vehicle navigation systems has brought the era of big data for the taxi industry. Existing destination prediction approaches are mainly based on various Markov chain models or trip matching ideas, which require geographical information and may encounter the problem of data sparsity. Other machine learning prediction models are still unsatisfying to provide favourable results. In this work, firstly, we propose a novel and efficient data embedding method for time-related features' pre-processing. The key idea behind this is to embed the data into a two-dimensional space before features learning. Secondly, we propose a novel ensemble learning approach for destination prediction. This approach combines the respective superiorities of support vector regression and deep learning at different segments of the whole

trajectory. Our experiments are conducted on two real data sets to exhibit the superior performance of our ensemble learning model. Comparisons also confirm the effectiveness of the proposed data embedding method in deep learning model.

In this thesis, a method based on bus trajectory data is developed for learning anomalous traffic patterns. Existing data-driven methods for traffic anomaly detection are modelled on taxi trajectory datasets. The concern is that the data may contain much inaccuracy about the actual traffic situations, because taxi drivers often choose optimal routes to evade from the congestions caused by traffic anomalies. We use bus trajectory data in this work. Bus trajectories can capture real traffic conditions in the road networks without drivers' preference, which are more objective and appropriate for accurately detecting anomalous patterns for a broad range of insight analyses on traffics. We propose a deep learning-based feature visualization method to map 3-dimensional features into a red-green-blue (RGB) color space. A color trajectory (CT) is then derived by encoding a trajectory with the RGB colors. With the spatial and temporal properties extracted from the CT, spatio-temporal outliers are detected by a novel offline detection method. We then conduct GIS map fusion to obtain insights for better understanding the traffic anomaly locations, and more importantly the influences on the road affected by the corresponding anomalies. Extended from the offline detection, an online detection method is developed for real-time detection of anomalous patterns. Our proposed methods are tested on 3 real-world bus trajectory datasets to demonstrate the performance of high accuracies, high detection rates and relatively low false alarm rates.

This thesis also introduces a novel deep learning-based model for urban traffic flow prediction. Accurate and reliable traffic flow prediction is a challenging task due to the highly non-linear and stochastic natures of traffic flow data, but its solutions are crucial for ITS. In this study, a novel deep learning approach is proposed to address the problem. Instead of utilizing grid search, we introduce a differential evolution

algorithm for globally optimizing the hyperparameters of an LSTM network, and parallel computing and early stopping criteria are implemented to accelerate the optimization process. The LSTM network with the optimized hyperparameters is then trained to learn sequential traffic flow features. The model is named parallel-differential-evolution-based LSTM network (PDE-LSTM). To the best of our knowledge, this is the first research that uses evolutionary algorithm to optimize deep learning models for traffic flow prediction. Experiments on three real-world traffic flow data sets from Dublin and San Francisco show that PDE-LSTM can achieve a high accuracy of at least 93% for all of the predictions. Comprehensive performance comparisons with state-of-the-art methods further confirm the superior performances of our deep learning approach on these real-world traffic flow data sets.



# Chapter 1

## Introduction

This chapter describes the background, research motivations, research objectives, research contributions and structure of the thesis. In Section 1.1, the backgrounds of traffic time series data, deep learning as well as some significant applications of traffic time series data analysis are presented. Section 1.2 introduces the motivations in this research work, including taxi destination prediction, anomalous traffic patterns detection and traffic flow prediction. The corresponding research objectives and contributions of each motivation are specified in Section 1.3. Finally, the structure of this thesis is detailed in Section 1.4.

### 1.1 Background

With the rapid deployment of intelligent transportation systems (ITS), huge amount of data stream from various traffic sensors has been generated and accumulated continuously. Among these data, traffic time series have been playing a crucial role in ITS research and applications (Li, Su, Zhang, Lin & Li 2015). Mining these traffic time series data can not only produce helpful information for relieving traffic problems such as congestion, but can also bring novel functions and services to ITS (Zhang, Wang, Wang, Lin, Xu & Chen 2011). However, due to the sparse, imbalanced, stochastic and highly

non-linear natures of traffic time series data, traditional parametric methods or machine learning methods are difficult to learn representations of data well. Recently, deep learning has attracted tremendous attention from researchers and brought about breakthroughs in image, video, speech, audio and text data processing (LeCun, Bengio & Hinton 2015). Therefore, considering the achieved milestones of deep learning techniques, this thesis aims to develop powerful and reliable deep learning-based data analysis functions to improve ITS service level and benefit stakeholders.

### 1.1.1 Traffic Time Series Data

A time series is formed by observations that have been collected over a fixed sampling interval. Time series data is commonly existed in almost every application fields in the world, such as business (e.g., sales figure), economics (e.g., stock prices), official statistics (e.g., census data), natural sciences (e.g., population size) and environmetrics (e.g., precipitation) (Dettling 2013). A time series process can be defined as a set of random variables  $\{x_t, t \in T\}$ , where random variable  $x_t$  is distributed according to some univariate probability distributions function  $P_t$ .  $T$  is a set of timestamps with equidistant time intervals, and  $T = \{1, 2, 3, \dots\}$ .

The time series data is ubiquitous in traffic with the rapid development of ITS. The methods of traffic time series data collection have also been evolving considerably. The collecting technologies of traffic time series data can be dichotomized into two families: the stationary and non-stationary sensor-based methods (Duan 2019, Leduc et al. 2008). The stationary sensor-based methods record data via detectors placed along the roadside or on the roads. This scenario basically includes pneumatic road tubes, magnetic loops, passive magnetic, microwave radar as well as ultrasonic and passive acoustic. The non-stationary sensor-based methods collect traffic data via detectors embedded on floating vehicles over the whole transport network. This methods include global positioning system (GPS) and cellular network. In Table 1.1, we give a brief description of the most important



Table 1.1: Sensor technologies of traffic time series data collection

Sensor	Description	Advantage	Limitation
Pneumatic road tubes	Rubber tubes are installed across the road lanes. When a vehicle tyre passes over the tube, the vehicle can be detected by picking up changes of pressure. They are usually used to measure the traffic flow.	Quick installation, low cost and simple to maintain.	Sensitive to weather, temperature and traffic conditions.
Magnetic loops	One of the most popular traffic sensors in ITS (Belenguer, Salcedo, Ibanez & Sanchez 2019). They are embedded in roadways to generate a magnetic field. The information is then transmitted to a counting device placed on the side of the road.	Function well even under bad weather conditions.	Short life expectancy, expensive cost for implementation and maintenance.
Passive magnetic	They are fixed under or on top of the roadbed, which are mainly used to count the number of vehicles, and to record their types and speeds.	Quick installation and high accuracy.	They are difficult to distinguish between closely spaced vehicles.
Microwave radar	They are used to detect moving vehicles and their speeds	Insensitive to inclement weather conditions, and available for multiple lane operation (U.S. Department of Transportation 2014).	Prone to false detection due to the presence of other objectives.
Ultrasonic and passive acoustic	They are installed alongside the road, and emit sound wave to identify vehicles by calculating the time of the signal back to them. It records vehicle number, speed and type.	Insensitive to precipitation (U.S. Department of Transportation 2014).	Can be affected by bad weather conditions.
GPS	It is a satellite-based radionavigation system. They are usually equipped in particular vehicles such as taxis and buses. It can record time, location, speed and heading.	It has low cost and high precision, and works well under any weather conditions.	The signal is easily obstructed by the obstacles like buildings.
Cellular network	It can approximate the actual coordinates of smartphone by tracking the mobile phone network from the smartphone equipped in the vehicle.	It does not require an active call, and easy to choose the positioning points of interest with the widespread use of smartphone (Duan 2019).	High cost and emerging privacy problem that is banned in some countries.

sensor technologies of traffic time series data collection.

### 1.1.2 Deep Learning

Before introducing deep learning, we shall start from the simple linear regression in statistics. Suppose we have  $N$  observations  $\{(x_1, y_1), \dots, (x_N, y_N)\}$ , variable  $\mathbf{x} = (x_1, x_2, \dots, x_N)$  is considered to be the independent variable, and  $\mathbf{y} = (y_1, y_2, \dots, y_N)$  is deemed to be the dependent variable. In linear regression, we assume that a linear relationship  $f(\mathbf{x}) = \mathbf{x}\mathbf{W} + \mathbf{b}$  is existed to map each  $x_i$  to  $y_i$ , and  $\mathbf{W}$  and  $\mathbf{b}$  are the parameters that determine the linear transformation with aim to minimize the mean squared error:  $\frac{1}{N} \|\mathbf{y} - (\mathbf{x}\mathbf{W} + \mathbf{b})\|_2^2$ .

In more general cases, inputs  $\mathbf{X}$  are composed by multiple independent variables, and so as the outputs  $\mathbf{Y}$ , where  $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m) \in \mathbb{R}^{N \times m}$ ,  $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_k) \in \mathbb{R}^{N \times k}$ ,  $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{iN})$  and  $\mathbf{y}_i = (y_{i1}, y_{i2}, \dots, y_{iN})$ . The relationship between inputs  $\mathbf{X}$  and outputs  $\mathbf{Y}$  may not be linear, we attempt to find a non-linear function  $f(\mathbf{X})$  to map  $\mathbf{X}$  to  $\mathbf{Y}$ . We assume that a non-linear transformation function  $\phi$  is defined. The inputs are firstly transformed by a linear function with parameters  $\mathbf{W}_1$  and  $\mathbf{b}_1$ . Then the outputs  $\mathbf{X}\mathbf{W}_1 + \mathbf{b}_1$  are fed into a non-linear transformation  $\phi$ , the feature vector is updated by  $\Phi(\mathbf{X}) = \phi(\mathbf{X}\mathbf{W}_1 + \mathbf{b}_1)$ . The feature vector via the non-linear transformation processing is fed again as the inputs of a linear function, then the model's outputs are written as  $f(\mathbf{X}) = \Phi(\mathbf{X})\mathbf{W}_2 + \mathbf{b}_2$ . The parameters  $\mathbf{W}_1$  and  $\mathbf{W}_2$  are matrices, and  $\mathbf{b}_1$  and  $\mathbf{b}_2$  are vectors. We can find the optimal  $\mathbf{W}_1$ ,  $\mathbf{W}_2$  as well as  $\mathbf{b}_1$ ,  $\mathbf{b}_2$  by minimizing the error between  $f(\mathbf{X})$  and  $\mathbf{Y}$  (Gal 2016).

A basic deep learning model can be viewed as a hierarchy of multiple non-linear processing aforementioned (Polson & Sokolov 2018). Each hierarchy is a level of representation, obtained by non-linearly transforming the representation at one level into another representation at a higher and more abstract level (LeCun et al. 2015). With the enough deep structures (enough such transformations) in a deep learning model, very complex relationship

could be learned over observations.

### 1.1.3 Applications of Traffic Time Series Data Analysis

Analysis of large amount of traffic time series data provides a new technical method for ITS. Stakeholders related to drivers, riders, traffic managers and transport service providers can benefit directly from various applications that provide users of convenience, safety as well as high-efficiency. Based on two relevant surveys from Zhu et al. (Zhu, Yu, Wang, Ning & Tang 2018) and Li et al. (Li, Su, Zhang, Lin & Li 2015), we will emphasize on 4 categories of traffic time series data analysis applications. However, with the rapid development of sensors, the following aspects cannot cover all those applications in this domain.

- **Location-based Services (LBSs)**

LBSs is a broad term that applications which utilize geographic data and information to provide services to users (Schiller & Voisard 2004). With the accurate geographic information provided from various navigation systems, usage of various LBSs has become a more and more important part in people's daily lives (Peng, Liu & Wang Mar. 2017). On one hand, LBSs can obtain the real-time position information of vehicle via GPS, and deliver real-time information, such as traffic condition, targeted advertising or activity recommending, to drivers and riders. On the other hand, instead of utilizing real-time location, LBSs can also use location further ahead by making prediction based on the historical movement data, such as the next location prediction (Chen, Liu & Yu 2014) and destination prediction (Zhang, Zhao, Zheng & Li 2019, Rossi, Barlacchi, Bianchini & Lepri 2020).

- **Traffic Anomaly Detection**

Anomaly detection refers to finding those behaviours that do not yield expected patterns. Traffic anomaly can be caused by traffic accidents, special events and loop detector faults (Li, Su, Zhang, Lin

& Li 2015). These abnormal patterns can also be propagated along the whole road networks, and they will not disappear automatically without proper traffic control strategies. Therefore, it is significant to develop a method to automatically figure out these abnormal patterns by data-driven techniques (Kong, Song, Xia, Guo, Wang & Tolba 2018, Xu, Ouyang, Cheng, Yu, Xiong, Ng, Pranata, Shen & Xing 2018). Detection of anomalous traffic flow pattern from continuous flow data can enable traffic managers to quickly respond to this changing situation. Anomalous traffic patterns detection from vehicles' GPS trajectory data also helps in sensing abnormal events and analyzing traffic accidents (Liu, Zheng, Chawla, Yuan & Xing 2011, Zhang, Zhang, Verma, Liu, Blumenstein & Li 2019).

- **Traffic Prediction**

Traffic prediction aims to characterise the relationship between the past traffic data and the future traffic data. Such prediction helps the users get a better understanding of the upcoming situation, and then prepare in advance. Accurate prediction of traffic speed over the whole transport network is helpful for route guidance and congestion avoidance (Asif, Dauwels, Goh, Oran, Fathi, Xu, Dhanya, Mitrovic & Jaillet 2013). Traffic flow prediction assists the users to make better travel decision and to guide traffic control strategies (Lv, Duan, Kang, Li & Wang 2014). Precise travel time prediction helps drivers and travellers to make decision or plan schedules (Wu, Ho & Lee Dec. 2004). Real-time prediction of travel time of taxi's trips is also beneficial for ridesharing and taxi dispatching in the taxi industry (Wang, Zheng & Xue 2014).

- **Asset Maintenance**

Proper maintenance approach of assets is essential to keep assets remain in a thriving condition and reduce maintenance costs. Analyzing the continuous performance data collected from vehicles or transport

infrastructures can help target problems at a faster and more accurate level. For example, time series data from vehicle or transport infrastructure, such as temperature, humidity, pressure, etc., can be collected via various sensors, and be processed and analyzed by the advanced data analysis methods. Then the current condition indicators of these assets can be diagnosed for further maintenance decisions making (Zhu et al. 2018).

## **1.2 Research Motivations**

### **1.2.1 Taxi Destination Prediction**

Taxi plays an important role in modern transport system all over the world (Bidasca & Townsend 2016, Ding, Liu, Pu & Ni 2013, Liu, Ni & Krishnan Jan. 2014). Over the past few decades, GPS has been widely used in a rapid increasing number of applications, such as vehicle based navigation system or smartphone based navigation system, which are broadly operated in the urban road network. Such a huge amount of movement data can be utilized in plentiful LBSs (Xu, Wang & Li 2016).

According to the statistics of the mobile searchers at a large software company, 68% of the searchers happened often in the transits, while 39% of them want to obtain the information about their destinations or near these destinations, and 12% of them want the information en route to their destinations (Teevan, Karlson, Amini, Brush & Krumm 2011). In the industry of taxi, accurate and timely destination prediction of taxis is of great importance for LBSs applications. For instance, application of targeted advertising, e.g., shopping, restaurants or hotels recommending, can be achieved via recommendation systems. Comparing to the existing advertising mode in taxi industry, there are significances of pertinence as well as high-efficiency. Meanwhile, real-time prediction of trips' destinations could also be helpful in taxi ride-hailing platforms like Uber, Grab or DiDi in the cases that the users alter their preset drop-off locations during services.

Existing methods of destination prediction are mainly based on trip matching, Markov chain models and machine learning models. Trip matching methods may not be efficient for huge volume of historical data. Markov chain models need supplementary geographical information and may encounter the data sparsity problem (Xue, Zhang, Zheng, Xie, Huang & Xu 2013). Other machine learning prediction models are still unsatisfying to learn features among limited prior knowledge, and are unable to achieve favourable prediction performances.

### 1.2.2 Anomalous Traffic Patterns Detection

Anomalous patterns detection from traffic data is of great significance in transportation. Detecting anomalous traffic patterns is to figure out unexpected patterns, which are helpful in traffic accidents analysis, fault detection, congestion management and new infrastructure planning (Li, Guo, Xia & Xie 2018). The anomalous traffic patterns can be reflected by investigating the trajectories of moving carriers in the road network (Liu et al. 2011). These patterns are emerged due to various factors including traffic accidents, traffic controls, parades, sports events, celebrations, disasters or other events. Existing methods for traffic anomaly detection are mainly modelled on city-wide taxi trajectory data (Liu et al. 2011, Chawla, Zheng & Hu 2012, Pang, Chawla, Liu & Zheng 2013, Wang, Lu, Yuan, Zhang & Van De Wetering 2013, Pang, Chawla, Liu & Zheng 2011, Wang, Wen, Yi, Zhu & Sun 2017, Mao, Sun, Jin & Zhou 2018, Zhang, Li, Zhou, Chen, Sun & Li 2011, Chen, Zhang, Castro, Li, Sun & Li 2011, Kuang, An & Jiang 2015, Yu, Cao, Rundensteiner & Wang 2014, Song, Wang, Xiao, Han, Cai & Shi 2018, Wu, Sun & Zheng 2017). However, the concern is that the data may contain much inaccuracy about the real traffic situations, because taxi drivers often choose optimal routes for themselves to evade from the congestions caused by various traffic anomalies (Kong et al. 2018). In this thesis, we will explore a more accessible trajectory data source of bus for probing the city-wide traffic anomalies. Bus services facilitate commuters

substantially. Take the city of Beijing as example, by the end of 2019, 23,010 buses have been on the roads everyday, serving 3.134 billion people with 1,162 regular bus routes during the year (Beijing Public Transport 2020). Bus services are available along the most major roads in metropolitan areas of city, and major roads are usually possessed with heavier traffic, which are much more meaningful for traffic research or management (Huang, Song, Hong & Xie 2014). In contrast to trajectory data of taxi, bus trajectory data for anomaly detection has the following advantages. Firstly, as a kind of public transport service, there is not much risk of privacy leakage regarding bus trajectory data. Secondly, as a result of such advantage, getting access to the real-time bus trajectory data becomes easy for many cities. This can be implemented via application programming interface (API) maintained by the traffic administrators. Thirdly, bus services have their own regular route, and the bus trajectory is more independent of the drivers' preference, reflecting more objectively on the actual road traffic conditions. This is contrast to taxi trajectory data which may lose much accuracy about traffic congestion situations, since taxi drivers can choose paths for themselves (Kong et al. 2018). Especially when a taxi driver gets the traffic information ahead, the driver very likely chooses an optimal route to avoid a foreseeable traffic congestion.

### **1.2.3 Traffic Flow Prediction**

Traffic flow prediction is aimed at making a prediction on the number of vehicles passing a specific observation point/region within a future time window (Huang et al. 2014, Pan, Sumalee, Zhong & Indra-Payoong 2013). Traffic flow prediction is a fundamental problem being addressed in ITS or smart city (Tian & Pan 2015). Accurate, reliable and timely traffic flow information is of great significance to the efficacy of various ITS subsystems, especially advanced transportation management systems, advanced traveler information systems, business vehicle management, and advanced public transportation systems, which have been all listed as the fundamental parts

of ITS (Zhang, Wang, Wang, Lin, Xu & Chen 2011).

Reliable prediction of traffic flow is still a challenge, because of the highly non-linear and stochastic natures of urban traffic flow data (Tian & Pan 2015). Recently, state-of-the-art deep learning methods have achieved superior performances in traffic flow prediction than other traditional parametric or machine learning methods (Qu, Li, Li, Ma & Wang 2019, Lv et al. 2014, Huang et al. 2014, Huang, Hong, Li, Hu, Song & Xie 2013, Tian & Pan 2015). However, hyperparameters optimization remains a tough problem in deep learning despite of its significant progresses. The deep learning-based traffic flow prediction methods mainly utilize grid search or random search strategies for hyperparameters tuning. However, grid search may lead to poor performance in practice since it is very computationally expensive, especially in the case of many hyperparameters or large sized sample data (Bergstra & Bengio 2012). Random search strategy reduces the computational cost significantly by scanning over a lower-dimensional subspace. However, the searched results may not be globally optimized.

### 1.3 Research Objectives and Contributions

To address above research motivations, this thesis focuses on 3 research problems: 1) taxi destinations prediction, 2) anomalous traffic patterns detection, and 3) traffic flow prediction. The **research objective** of this thesis is to contribute to ITS applications by developing novel deep learning-based approaches that could generate accurate, reliable and robust models on traffic time series data. The specific objectives of above research problems are summarized as follows (O1 to O3).

- O1: To develop a data-driven ensemble learning approach to accurately predict taxi destinations.
- O2: To develop offline and online methods for traffic anomaly detection from trajectory features extracted by deep architecture, and to develop an insight analysis method for better understanding traffic anomaly.



O3: To develop an evolutionary algorithm-based deep learning approach for accurate and robust prediction of traffic flow.

To complete these objectives, we have proposed 3 novel methods as presented in Chapter 3, Chapter 4 and Chapter 5, respectively. Our **contributions** are elaborated as follows (C1 to C3).

**C1: Taxi destinations prediction from trajectories using a novel data embedding method and ensemble learning**

The contributions include : 1) A novel and efficient data embedding method is proposed for time-related features embedding. 2) We develop a data-driven ensemble learning approach for destination prediction, combining the respective superiorities of support vector regression (SVR) and deep learning at different segments of the trajectory. 3) We conduct extensive experiments on two real-world datasets to confirm the superior prediction performance as well as the effectiveness of proposed data embedding method.

**C2: Offline and online detection of anomalous patterns from bus trajectories for traffic insight analysis**

Our contributions in this research are summarized : 1) We present a deep neural network architecture to extract deeply hidden features for generating better features visualization than typical dimensionality reduction methods, and conduct GIS fusion for getting insights into the anomalies, for example, the anomaly locations and their impacts caused to the road traffic. 2) We devise a novel method for an offline detection of anomalous traffic patterns at bus route level. Particularly, instead of introducing machine learning models, we design algorithm on imbalanced data by addressing the discrepancy between different classes of anomaly. 3) Extended from the feature extraction architecture and the offline detection method, we propose an online method for real-time detection of anomalous traffic patterns. 4) We perform comprehensive experiments on three real-world datasets to confirm the effectiveness

and superiority of the deep feature extraction architecture, the offline and online anomaly detection methods and insight analysis of the anomalous patterns.

### **C3: Differential evolution based LSTM recurrent neural network for traffic flow prediction**

The contributions in this research include : 1) This is the first work to use evolutionary algorithms to optimize deep learning model for traffic flow prediction. 2) Parallel computing and early stopping strategy are implemented in the model to accelerate the optimization process. 3) We conduct experiments on three real-world traffic flow datasets from different cities to demonstrate the high prediction accuracy of our proposed model. Comparisons with the state-of-the-art methods further confirm the superior performances of our deep learning approach.

## **1.4 Thesis Structure**

The structure of this thesis is illustrated in Figure 1.1, and it is briefly introduced as follows:

**Chapter 1** introduces the background of this thesis, the research motivations and the corresponding research objectives as well as contributions. **Chapter 2** presents the related work of this research, including taxi trajectory modelling, anomalous patterns detection and traffic flow prediction. **Chapter 3** to **Chapter 5** detail the proposed methods on taxi destination prediction, anomalous traffic patterns detection and traffic flow prediction, respectively. Details of experimental evaluation, comparison and analysis are also included. **Chapter 6** concludes this thesis and provides discussions of future work.

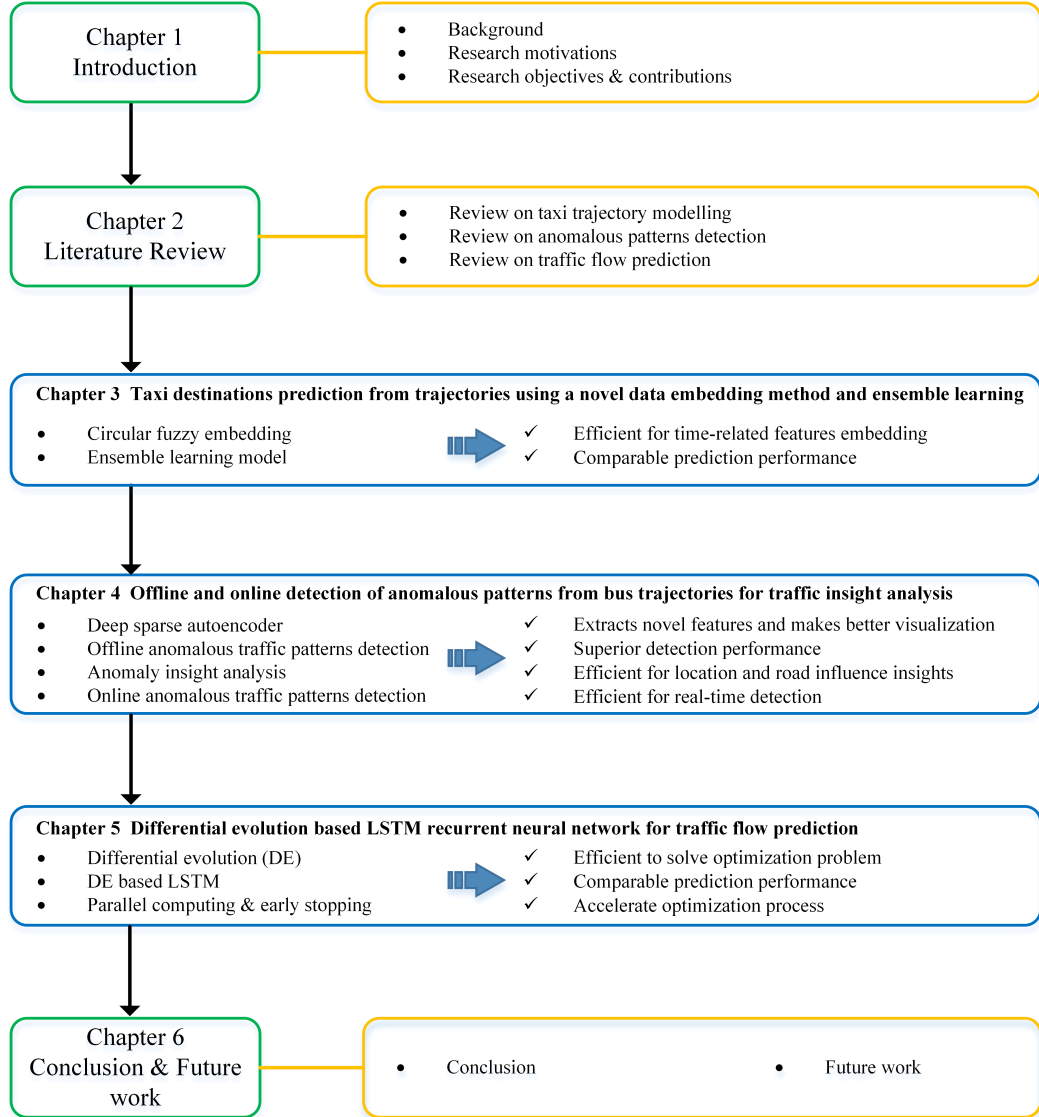


Figure 1.1: Thesis structure.

## Chapter 2

# Related Work and Literature Review

This chapter describes the related work and literature review of the work in this thesis. Section 2.1 reviews the relevant work on taxi trajectory modelling. Then, the state-of-the-art methods for anomalous patterns detection are presented in Section 2.2. Following this, the existing methods for traffic flow prediction are introduced in Section 2.3. Finally, we briefly summarize the contents in this chapter.

### 2.1 Taxi Trajectory Modelling

The analysis of taxi trajectory datasets has been considered by a lot of researches in the subjects of data mining, machine learning or intelligent transportation systems. Recommendation systems and location prediction are two popular topics among them that utilize taxi trajectory datasets.

#### 2.1.1 Recommendation Systems

The taxi trajectory-based recommendation systems mainly include routing recommending (Liu & Qu 2016, Dai, Yang, Guo & Ding 2015, Dai et al. 2015), passenger-hunting recommending (Ding et al. 2013), taxi-hunting

recommending (Xu, Zhou, Liu, Xu & Zhao 2014), social recommending (Liu & Wang 2017) and charging station recommending for electric vehicles (EV) (Tian, Jung, Wang, Zhang, Tu, Xu, Tian & Li 2016).

Liu et al. (Liu & Qu 2016) proposed a dynamic congestion conditions prediction framework using topic-aware Gaussian process, then adaptive routing recommendation algorithm was applied. This framework is not only limited to GPS trajectory data, it can also be extended to traffic data from road sensor system. A personalized route recommendation method was presented by capturing the taxi drivers' driving preferences from the taxi trajectories (Dai et al. 2015). It is the first work to use big trajectory data for personalized route recommendation. Ding et al. (Ding et al. 2013) developed a passenger-hunting system to recommend a connected trajectory with the objective to produce higher profit. Study by Xu et al. (Xu et al. 2014) proposed a taxi-hunting recommendation system to estimate the probability and waiting time in a particular location. It combines an offline processing phase and a fast online inquiring phase based on the probability model. Liu et al. (Liu & Wang 2017) developed a community detection technique based on mobility trajectory, then an online recommendation method based on trajectory community was proposed to improve service level. In the study by Tian et al. (Tian et al. 2016), a real-time charging station recommendation system was presented for EV taxis by mining large-scale GPS data to save the most time. This is the first recommendation system for EV taxis.

### 2.1.2 Location Prediction

Location prediction of a moving objective has been one of the most traditional problems in trajectory analysis. A common and simple approach is trip matching, if an on-going trip (i.e., query trip) matches part of a popular trajectory from the historical trajectories, then the destination of this popular route will be taken as the destination of the query trip (Xue et al. 2013). However, trip matching might not be efficient on huge amount of data. Other methods are mainly based on Markov chain models (Ashbrook &

Starner Sept. 2003, Li, Ahmed & Smola 2015, Simmons, Browning, Zhang & Sadekar 2006, Alvarez-Garcia, Ortega, Gonzalez-Abril & Velasco Dec. 2010, Ziebart, Maas, Dey & Bagnell 2008). Firstly, Ashbrook and Starner introduced Markov chain model to predict the most likely next location in 2003. This model consists of nodes, each node represents a location, which are used as the states of Markov process. Then, the probabilities can be derived from the historical locations which have been visited. The transition between two states represents the probability of the user travelling between these two locations, which can be trained through the historical trajectories (Ashbrook & Starner Sept. 2003). Simmons et al. (Simmons et al. 2006) used a hidden Markov chain model (HMM) to predict the route and destination of the driver based on an online observation of their GPS position. Several variants of Markov chain model (Gambs, Killijian & del Prado Cortez 2012, Alvarez-Garcia et al. Dec. 2010) were also presented in location prediction based on their past GPS log data. However, these approaches need to be combined with extra geographical information, such as GIS map database, which aims to provide road graph consisting of road intersections and linking between intersections (Simmons et al. 2006). Sometimes they may lead to data sparsity problem in practice as the historical trajectory data cannot cover all possible query trajectories (Xue et al. 2013), i.e., the query trajectory does not match any historical trajectory or the probability of the transition between two locations approximates zero.

Machine learning methods have also been applied for predicting locations. Artificial neural network (ANN) with shallow structure was introduced in taxi destination prediction (De Brébisson, Simon, Auvolat, Vincent & Bengio 2015). The input layer of this model are the initial and last points of the historical trajectory prefix integrated with some meta-data embedding, such as client ID, taxi ID, stand ID and time information. The output layer are the clusters of corresponding destinations. Following this, state-of-the-art machine learning models like decision tree (Manasseh & Sengupta 2013, Costa, Fontes, Costa & Dias 2015), bootstrapped decision tree, decision tree

with pruning (Manasseh & Sengupta 2013), naive Bayes (Costa et al. 2015), reinforcement learning (Le, Liu & Lau 2016) and recurrent neural network (RNN) (Rossi et al. 2020) have also been developed in location prediction.

## 2.2 Anomalous Patterns Detection

Anomalous pattern detection aims to detect the unexpected patterns, which has been intensively studied in the domain of data mining and knowledge discovery (Liu et al. 2011). Firstly, the methods for detecting general anomalies/outliers are reviewed in Section 2.2.1, four basic categories of methods are presented. Then, Section 2.2.2 conducts a comprehensive review on anomaly detection in the road traffic domain.

### 2.2.1 General Anomaly Detection

Anomalous pattern detection aims to detect unexpected patterns, which has been intensively studied in the domain of data mining and knowledge discovery (Liu et al. 2011). To our best knowledge, at least four categories of methods were proposed, including dimensionality reduction-based methods (Lakhina, Crovella & Diot 2004, Liu, Zhang & Guan 2010, Callegari, Gazzarrini, Giordano, Pagano & Pepe 2011, Juvonen & Hamalainen 2014, Fontugne, Abry, Fukuda, Borgnat, Mazel, Wendt & Veitch 2015, Sakurada & Yairi 2014), unsupervised methods (Münz, Li & Carle 2007, Leung & Leckie 2005, Pawling, Chawla & Madey 2007, Li, Huang, Tian & Xu 2003, Wang, Wong & Miner 2004, Zhang, Song, Chen, Feng, Lumezanu, Cheng, Ni, Zong, Chen & Chawla 2019, Lv, Yu, Fan, Tang & Tong 2020), supervised classification-based methods (Hautamaki, Karkkainen & Franti 2004, Song et al. 2018, Malhotra, Ramakrishnan, Anand, Vig, Agarwal & Shroff 2016, Chauhan & Vig 2015, Kim & Cho 2018) and statistical methods (Barbará, Domeniconi & Rogers 2006, Fan & Xiong 2013, Rogers, Barbará & Domeniconi 2009).

Dimensionality reduction method like principal component analysis

(PCA) has been validated effective in anomalous patterns detection (Lakhina et al. 2004, Liu et al. 2010). In (Callegari et al. 2011), an improved PCA by introducing Kullback-Leibler divergence was proposed for network anomaly detection. Random projection (RP) was used for dimensionality reduction and to detect internet traffic anomaly (Juvonen & Hamalainen 2014, Fontugne et al. 2015), since it is very fast and can perform even in real-time. Apart from linear methods, a nonlinear dimensionality reduction method (autoencoders) has also been presented in (Sakurada & Yairi 2014).

Unsupervised clustering algorithms including k-means (Münz et al. 2007), density-based and grid-based clustering (Leung & Leckie 2005) and one pass clustering (Pawling et al. 2007) were proposed to identify anomalous network patterns. Besides clustering algorithms, one-class support vector machine (OneSVM) with novel kernels was introduced to detect malicious intrusion to computer systems (Li et al. 2003, Wang et al. 2004). Recently, unsupervised deep learning-based methods have also been presented for modelling large scale data, and to detect anomalies (Zhang, Song, Chen, Feng, Lumezanu, Cheng, Ni, Zong, Chen & Chawla 2019, Lv et al. 2020).

Supervised classification-based methods include k-nearest neighbour (kNN), recurrent neural network (RNN) and long short-term memory (LSTM). Literature (Hautamaki et al. 2004) sorted the average kNN distances in ascending orders, and then outliers were defined when the difference between two nearby distances is greater than a preset threshold (Hautamaki et al. 2004). Besides the lazy learning approach, recently, supervised deep learning-based anomaly detection methods have also contributed to solve this problem, including RNN-based model (Song et al. 2018) and LSTM-based model (Malhotra et al. 2016, Chauhan & Vig 2015, Kim & Cho 2018).

There are also some studies introducing statistical methods for outlier detection. Barbará et al. (Barbará et al. 2006) proposed to use transductive confidence machines and hypothesis testing to uncover outliers. It only has two parameters, and neither of them requires careful tuning. Fan



et al. (Fan & Xiong 2013) presented a privacy-preserving framework for anomaly detection based on continual aggregate statistics. It enables real-time detection and provides privacy guarantee. In (Rogers et al. 2009), a multi-modal distance measure was defined to evaluate the strangeness. Furthermore, statistical testing was applied to estimate the probability of anomaly.

### 2.2.2 Road Traffic Anomaly Detection

Since the anomalous patterns in road traffic possess their own characteristics, some particular methods were presented on the top of aforementioned general anomaly detection methods. Based on the data source utilized, they could be dichotomized into two families: by using trajectory data sources or by using other data sources.

Apart from trajectory data sources, other data sources used for road traffic anomaly detection are mainly non-structured, which include social media data (Nguyen, Liu, Rivera & Chen 2016), video surveillance data (Li et al. 2018, Barria & Thajchayapong 2011, Li, Liu & Huang 2016, Zhao, Yi, Pan, Zhao, Zhao, Su & Zhuang 2019) or heterogeneous traffic data (Riveiro, Lebram & Elmer 2017). Literature (Nguyen et al. 2016) used text data from Twitter for real-time traffic incident detection. Li et al. (Li et al. 2018, Barria & Thajchayapong 2011, Li et al. 2016, Zhao et al. 2019) employed video data collected from traffic surveillance cameras to detect or classify traffic anomalies. In addition to using single data source, study by Riveiro et al. (Riveiro et al. 2017) explored the heterogeneous data sources from various vehicle embedded sensors for traffic anomaly detection.

Trajectory-based road traffic anomaly detection has been intensively investigated by many studies, while most of which are based on city-wide taxi trajectories. Studies by (Chawla et al. 2012, Kuang et al. 2015) used PCA or wavelet transform technique to identify traffic anomalies from taxi trajectory data. In (Pang et al. 2013) and (Pang et al. 2011), likelihood ratio test was introduced to represent traffic patterns and to detect anomalous patterns.

It has demonstrated accurate and fast detection on real data sets. Liu et al. (Liu et al. 2011) constructed an anomaly detection model by building a region graph, where a node represents a region and the link between every two nodes denotes the traffic flow, and then the extreme outliers could be detected from the graph links. In (Wang et al. 2017), tensor decomposition technique was employed for learning dynamic context features from taxi traces data, and then anomalous degrees for road segments were calculated. Authors in (Yu et al. 2014) proposed neighbor-based trajectory outlier definitions, and designed an optimized strategy to detect new outlier classes from massive-scale trajectory streams. In (Mao et al. 2018), a feature grouping-based anomaly detection framework was proposed to identify outliers from distributed trajectory streams. Study work by Wang et al. (Wang et al. 2013) estimated traffic flow speed on the road, and then traffic jam events were automatically detected based on relative low road-speed detection. Research (Zhang, Li, Zhou, Chen, Sun & Li 2011) demonstrated a method to group taxi trajectories crossing the same source destination cell-pair, then isolation mechanism was employed to detect abnormal trajectory. Wu et al. (Wu et al. 2017) developed a novel outlier detection approach by modeling the human driving behavior from historical taxi trajectories. This is the first work that combines human driving behavior modelling into outlier detection.

## **2.3 Traffic Flow Prediction**

Traffic flow prediction has been intensively investigated in many studies, because of its importance to ITS implementation. Overall, based on the methods utilized, the literature work can be dichotomized into two families: the parametric methods (Section 2.3.1) and nonparametric methods (Section 2.3.2).

### 2.3.1 Parametric Methods

The auto-regressive integrated moving average (ARIMA), a traditional parametric method, has been employed to construct models for short-term forecasting of traffic flow (Ahmed & Cook 1979). Many upgraded models based on ARIMA were also proposed to improve the performances of flow prediction, including the seasonal ARIMA model (Kumar & Vanajakshi 2015), the space-time ARIMA model (Lin, Huang, Zhu & Wang 2009, Ding, Wang, Zhang & Sun 2011), the subset ARIMA (Lee & Fambro 1999), the vector auto-regressive moving average (VARMA) approach (Min & Wynter 2011) and so on. Integration studies of ARIMA with exponential smoothing (ES) for traffic flow forecasting were investigated by (Van Der Voort, Dougherty & Watson 1996) and (Tan, Wong, Xu, Guan & Zhang 2009). Apart from the ARIMA-based parametric methods, Kalman filtering (Guo, Huang & Williams 2014) and chaotic time series analysis (Jieni & Zhongke 2008) have been developed in traffic flow prediction applications as well.

### 2.3.2 Non-parametric Methods

Due to the highly non-linear stochastic natures of urban traffic flow, parametric methods cannot depict it precisely with the quite limited distributional assumptions. Therefore, nonparametric methods have gained more attentions over the recent decades. Literature work by Davis et al. (Davis & Nihan 1991) used kNN for freeway traffic flow regression and conjectured that larger datasets might get better performances using this method. Literature works (Yang, Tan, Wang, Tian & Pan 2010, Hong, Dong, Zheng & Lai 2011, Castro-Neto, Jeong, Jeong & Han 2009) have proposed to employ supervised learning method of SVR for traffic flow forecasting. Besides SVR, another kind of kernel-based machine learning model, Gaussian process (GP), was used in studies (Xie, Zhao, Sun & Chen 2010, Sun & Xu 2010, Zhao & Sun 2016) for traffic flow regression. Sun et al. (Sun, Zhang

& Yu 2006) provided a Bayesian network for modelling traffic flows, by which the joint probability distribution between the cause nodes (flow data utilized for prediction) and the effect node (flow to be predicted) is formulated as a Gaussian mixture model (GMM). A Bayesian combination method (BCM) was proposed to integrate three individual predictors to improve the short-term traffic flow forecasting performance (Wang, Deng & Guo 2014).

Besides these non-neural models aforementioned, various neural network (NN) based models (Dougherty & Cobbett 1997, Hodge, Krishnan, Austin, Polak & Jackson 2014, Zhu, Cao & Zhu 2014) and fuzzy neural network (FNN) based models (Yin, Wong, Xu & Wong 2002, Li 2016, An, Fu, Hu, Chen & Zhan 2019) have been developed for forecasting traffic flow.

Recently, deep learning has been intensively involved in the field of traffic data analysis due to its remarkable representation capability. Lv et al. (Lv et al. 2014) employed a stacked autoencoder (SAE) model for traffic flow features learning and achieved a significant improvement in terms of forecasting accuracy. An SAE Levenberg-Marquardt (SAE-LM) model was proposed to further improve the performance using the Taguchi method to optimize the network structure (Yang, Dillon & Chen 2016). As suggested by the literature (Huang et al. 2014, Huang et al. 2013), a deep belief network (DBN) is able to learn features very well from limited prior knowledge for accurate traffic flow forecasting. The work by Qu et al. (Qu et al. 2019) developed a long-term traffic flow prediction model by training a deep neural network (DNN) with historical traffic flow data and contextual factor data. The study (Tian & Pan 2015, Kang, Lv & Chen 2017) trained an LSTM network for traffic flow prediction, and it has been demonstrated that the model can outperform most of the nonparametric models aforementioned on real-world datasets. Recently following similar ideas, the work by Luo et al. (Luo, Li, Yang & Zhang 2019) presented a hybrid model from kNN and LSTM for spatio-temporal traffic flow forecasting. Besides, various novel deep learning models based on LSTM and convolutional neural network (CNN) were proposed to learn the spatio-temporal characteristics of traffic

flow (Wu & Tan 2016, Mihaita, Li, He & Rizoiu 2019). Another useful work (Tian, Zhang, Li, Lin & Yang 2018) concentrates on traffic flow prediction with missing data which takes a novel LSTM-based method to deal with the missing patterns. Very recently, the research (Yang, Sun, Li, Lin & Tian 2019) provided an attention-mechanism-based LSTM network by enhancing traffic flow features to capture the high-impact flow values.

## **2.4 Summary**

In this chapter, a comprehensive literature review has been conducted with respect to the research motivations of taxi destination prediction, anomalous traffic patterns detection and traffic flow prediction. More precisely, relevant studies on recommendation systems and location prediction modelled on taxi trajectory data are described, followed by the methods of anomalous patterns detection, including general anomaly detection methods and methods specialized on road traffic anomaly detection. Finally, typical parametric and non-parametric methods for urban traffic flow prediction are reviewed.

## Chapter 3

# Taxi Destinations Prediction from Trajectories Using a Novel Data Embedding Method and Ensemble Learning

In this chapter, we attempt to unite multiple machine learning methods for destination prediction that learns features among limited prior knowledge. More specifically, an ensemble learning model (ELM) based on support vector regression (SVR) and deep learning (deep belief network (Hinton, Osindero & Teh May 2006)) is proposed. Specifically, these two models perform better than others at different segments of the whole trajectory. For the architecture in deep learning, we propose a novel data embedding technique named circular fuzzy embedding (CFE) for time-related features representation, which maps high-dimensional data into a two-dimensional space. Finally, experiments conducted on two independent real-world datasets demonstrate that our proposed ensemble learning model for destination prediction has superior performance comparing with the existing methods.

The organization of this chapter is structured as follows. Section 3.1 elaborates the proposed method of CFE and ELM for destination prediction.

Section 3.2 presents the experiments and analysis of the results. Section 3.3 summarizes this chapter.

## **3.1 Methods**

In this section, we first introduce a novel data embedding technique called circular fuzzy embedding (CFE) for representing time-related features before features learning. It maps higher dimensional feature into a two-dimensional space, and fuzzy membership is introduced to avoid the instability between the adjacent sections. Then, an ensemble learning model (ELM) based on SVR and deep belief network (DBN) for destination prediction is proposed. SVR is an efficient supervised learning method that has been applied widely in pattern recognition. The basic principle is to transform the training data from the input space into a hyper feature space by a kernel function, and then to find an optimal regression function by minimizing the regression loss (Wu et al. Dec. 2004, Smola & Schölkopf Aug. 2004). DBN is a stack of restricted Boltzmann machine, each one has one layer of hidden units and one layer of visible units, where unsupervised pre-training is employed before fine-tuning (Huang et al. 2014, Hinton Mar. 2002). A restricted Boltzmann machine is an undirected graphical model that visible units are connected with hidden unit via undirected weighted connections (Teh & Hinton 2001), while there is no visible-visible units or hidden-hidden units connection (Mohamed, Dahl & Hinton Jan. 2012).

### **3.1.1 Circular Fuzzy Embedding (CFE)**

In the domain of transportation, some types of time-related data are discrete, such as date, days of the week, day type. For discrete data processing method in machine learning, the most common one is one-hot embedding technique, which converts discrete features into binary vectors. For example, supposing we have a three-categorical feature comprising of “Holiday”, “Weekday” and “Weekend”, the feature of “Holiday” can be converted into binary vector

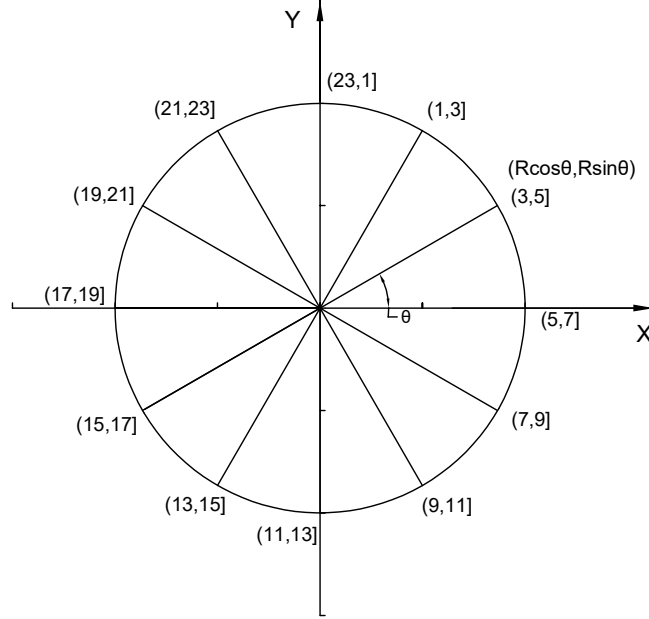


Figure 3.1: Circle constructed for feature of hour of day.

of  $(1, 0, 0)$ . Similarly,  $(0, 1, 0)$  and  $(0, 0, 1)$  correspond to categories of “Weekday” and “Weekend”, respectively. However, such embedding technique might have the following drawbacks:

1. May lead to data sparsity and curse of dimensionality (Wang, Xu, Xu, Tian, Liu & Hao Jan. 2016);
2. Occupy large memory usage if the size of category is huge;
3. Slow down training of network with large number of category;
4. Consider little about similarities between observations. As it turns into binary vector, the similarities between any two observations are the same.

To avoid these phenomenons, we propose a novel technique named CFE for time-related date embedding. The ideas of CFE technique comes from



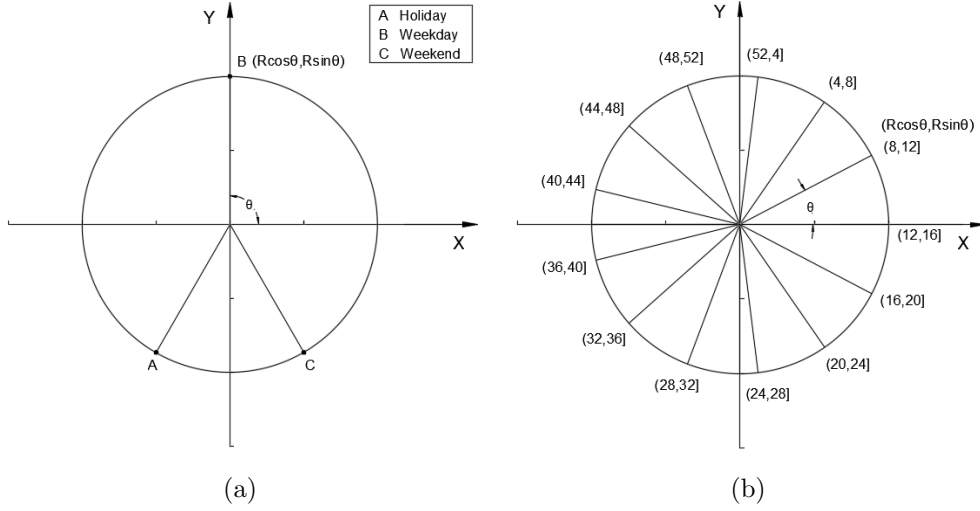


Figure 3.2: (a) Circle constructed for feature of day type. (b) Circle constructed for feature of week of year.

Word2vec, which is a frequently used model to generate word embeddings (Mikolov, Chen, Corrado & Dean 2013).

The first step is to construct a circle centred on  $(0, 0)$  to embed all the categories of feature. The reason for using a circle is because it not only represents the unique identity but also measures the similarities easily. However, for those continuous variables, firstly, we convert them into discrete sections. For instance, we embed feature of hour of day on a circle averagely. Firstly, we divide a whole day (24 hours) into 12 discrete and disjoint ranges, from  $(23, 1]$  to  $(21, 23]$ , which has been demonstrated in Fig. 3.1. Supposing the radius denotes  $R$ , the radian from each category to the X-axis represents  $\theta$ , then each category could be represented by a coordinate in a two-dimensional space. Compared with the twelve-dimensional space of one-hot embedding, it reduces the dimensionality significantly. In addition, it also considers about the difference of similarities between categories, because traffic in the adjacent time periods are more likely to have similar patterns in the domain of transport (Peng, Jin, Wong, Shi & Liò Apr. 2012, Liu, Gong, Gong & Liu Feb. 2015). For example, in Fig. 3.1, the travelling pattern

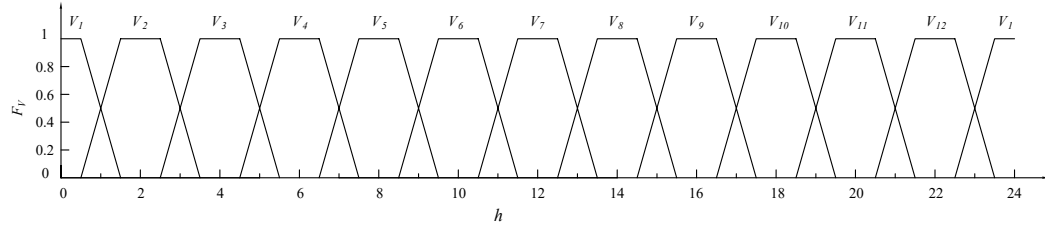


Figure 3.3: Linear combined membership function of hour of day feature.

in the range of  $(23, 1]$  is more likely to be similar with periods between  $(1, 3]$  or  $(21, 23]$  than range  $(11, 13]$ . Similarly, as shown in Fig. 3.2, we embed features of day type and week of year (52 weeks) on other two circles, respectively. In Fig. 3.2 (a), we address the issue that the differences between categories of “Holiday”, “Weekday” and “Weekend” are not the same.

As shown in Fig. 3.1, different sections of time are embedded into two-dimensional space with unique vectors. However, time around the bounds of the adjacent sectors may be embedded with quite different vectors, while there may be little difference between them in fact. For instance, time range  $(1, 5]$  is divided into two sectors,  $(1, 3]$  and  $(3, 5]$ , each with two-hour interval. Sector  $(1, 3]$  is embedded into vector of  $\mathbf{q}_2$  and sector  $(3, 5]$  into  $\mathbf{q}_3$ ,

$$\mathbf{q}_2 = (R \cos(\pi/3), R \sin(\pi/3)) \in \mathbb{R}^2 \quad (3.1)$$

$$\mathbf{q}_3 = (R \cos(\pi/6), R \sin(\pi/6)) \in \mathbb{R}^2 \quad (3.2)$$

Supposing we have time points of 02:59:00 and 03:01:00, they are embedded into two totally different vectors because they are located in different sectors. However, the difference between them is very tiny. In order to avoid such unstable situation, we introduce the membership function in fuzzy set theory (Zadeh June 1965). The membership function ( $F_V$ ) of hour of day ( $h$ ) is illustrated in Fig. 3.3,  $V_1, V_2, \dots, V_{12}$  denote the time sectors  $(21,1], (1,3], \dots, (21,23]$ , respectively.

The membership functions  $F_{V_1}(h)$ ,  $F_{V_2}(h)$  and  $F_{V_3}(h)$  for hour sectors  $(23, 1]$ ,  $(1, 3]$  and  $(3, 5]$  can be written as Eq. (3.3), Eq. (3.4) and Eq. (3.5), respectively.

$$F_{V_1}(h) = \begin{cases} h - 22.5, & 22.5 \leq h < 23.5 \\ 1, & 0 \leq h < 0.5 \text{ or } 23.5 \leq h < 24 \\ 1.5 - h, & 0.5 \leq h < 1.5 \\ 0, & \text{others} \end{cases} \quad (3.3)$$

$$F_{V_2}(h) = \begin{cases} h - 0.5, & 0.5 \leq h < 1.5 \\ 1, & 1.5 \leq h < 2.5 \\ 3.5 - h, & 2.5 \leq h < 3.5 \\ 0, & \text{others} \end{cases} \quad (3.4)$$

$$F_{V_3}(h) = \begin{cases} h - 2.5, & 2.5 \leq h < 3.5 \\ 1, & 3.5 \leq h < 4.5 \\ 5.5 - h, & 4.5 \leq h < 5.5 \\ 0, & \text{others} \end{cases} \quad (3.5)$$

In the same way, the membership functions for the rest sectors can also be derived. Then, we can get the final embedding vectors ( $\mathbf{b}_1(h)$ ) of the hour of day feature, as shown by

$$\mathbf{b}_1(h) = \sum_{i=1}^{12} F_{V_i}(h) \cdot \mathbf{q}_i \quad (3.6)$$

where  $\mathbf{q}_i$  represents the  $i$ th embedding vector before introducing fuzzy membership,  $F_{V_i}(h)$  denotes the membership of time point  $h$  corresponding to  $V_i$ . With our proposed CFE method aforementioned, we can also get the embedding vectors of day type feature  $\mathbf{b}_2$  and week of year feature  $\mathbf{b}_3$ .

### 3.1.2 Ensemble Learning Model (ELM)

The key idea behind ELM is to construct a knowledge base, and apply different models in the knowledge base under different conditions to get a superior prediction. In this work, the knowledge base is formed by models of SVR and DBN, as they perform better than others in different proportions of the whole trajectory of taxi. Specifically, it gets better prediction performance with DBN (classification) when the taxi is currently located in the initial part of the whole trip, otherwise, SVR does better. The elaborate experimental results can be referred to Fig. 3.6, Table S1 and Table S2 in Supplementary file 1. Therefore, the key of ensemble mechanism turns into how to detect which proportion of the whole on-going trajectory the taxi is currently located at. So that the best model could be allocated accurately.

---

#### Algorithm 3.1 Algorithm of ELM

---

**Constant Parameters:**  $\theta_{SVR}$ ,  $\theta_{DBN}$ ,  $\lambda$ ,  $k$ .

**Input:** input trajectory for training  $\mathbf{X} = (\mathbf{a}_i) \in \mathbb{R}^T$ ,  $\mathbf{a}_i \in \mathbb{R}^{\text{length}(a_i)}$ , output target for training set  $\mathbf{Y}$ , input trajectory for prediction  $\mathbf{X}^* = (\mathbf{x}_p) \in \mathbb{R}^\Gamma$ ,  $\mathbf{x}_p \in \mathbb{R}^{\text{length}(x_p)}$ .

**Output:** predicted GPS coordinate  $\mathbf{o}$ .

```

1:  $\mathbf{l} = () * T$ ,  $\mathbf{label} = (0) * T$ ;
2: Train a SVR model  $S$  using  $\mathbf{X}$ ,  $\mathbf{Y}$  and hyperparameters  $\theta_{SVR}$ ;
3: Train a DBN model  $D$  using  $\mathbf{X}$ ,  $\mathbf{Y}$  and hyperparameters  $\theta_{DBN}$ ;
4:  $m = -1$ ;
5: for  $i = 0$ ;  $i < T$ ;  $i++$  do //construct a classifier to estimate the current
   proportion of the whole trajectory
6:    $m++$ ;
7:   for  $j = 0.1$ ;  $j \leq 0.95$ ;  $j+ = 0.05$  do //extract input trajectory with
   the proportion from 10% to 95%
8:      $s = \text{round}(0.5 * \text{length}(\mathbf{a}_i) * j) * 2$ ;
9:      $\mathbf{l}(m) = \mathbf{a}_i[0 : s]$ ;
10:    if  $j > \lambda$  then
```

```

11:         label( $m$ ) = 1; //binary labelling for this trajectory
12:     end if
13: end for
14: end for
15: for  $p = 0; p < \Gamma; p++$  do
16:      $\rho = kNN(\mathbf{l}, \mathbf{label}, \mathbf{x}_p, k)$ ; //apply kNN to estimate the current
        proportion
17:     if  $\rho = 1$  then
18:          $\mathbf{o} = S(\mathbf{x}_p)$ ;
19:     else
20:          $\mathbf{o} = D(\mathbf{x}_p)$ ;
21:     end if
22: end for

```

---

Algorithm 3.1 illustrates our ELM algorithm. Define  $\mathbf{X}$  and  $\mathbf{Y}$  as the input trajectory data and target data.  $\mathbf{X}^*$  denotes the input trajectories for prediction and  $\mathbf{o}$  is the predicted output. Step 2 and Step 3 train each model with hyperparameters  $\theta_{SVR}$  and  $\theta_{DBN}$ , respectively. Steps from 4 to 14 is the key component of our proposed ELM, it constructs the training data and corresponding labels for segment estimation classifier, which provides basis for model selection. From step 4 to 9, the input trajectory prefix  $\mathbf{X}$  is extracted with the prior proportions (from 10% to 95%, with increment of 5%) of the whole trajectory. Steps 10 to 12 set the corresponding target **label** = 1 only if the prior proportion of the extracted trajectory exceeds  $\lambda$ , which is a constant parameter based on the performance of SVR and DBN, otherwise, **label** = 0. Steps from 15 to 22 elaborate the mechanism of proposed ELM. In step 16, a lazy supervised learning approach  $k$ -nearest neighbor (kNN) algorithm is applied to predict the segment proportions of the current position with query trajectory  $\mathbf{X}^*$ , as it is a simple and fast algorithm that performs well on large sized training data. Finally, SVR and DBN can be applied based on the binary label of  $\rho$ . Fig. 3.4 gives the brief workflow of ELM.

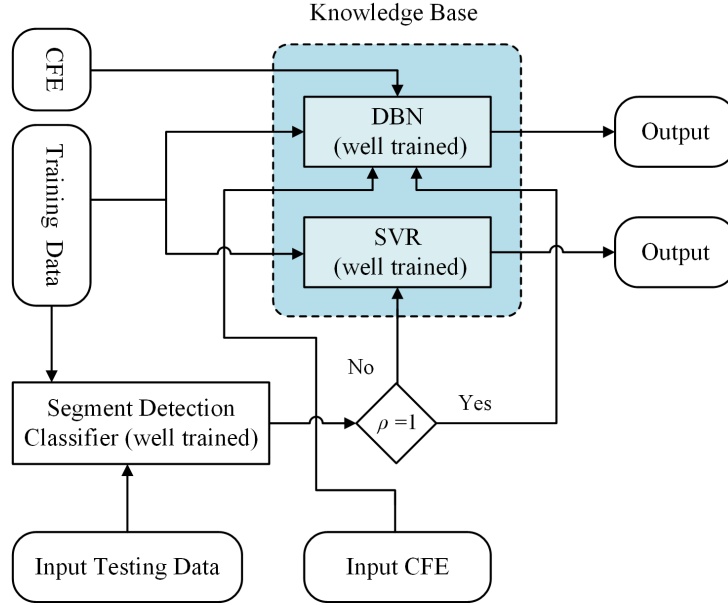


Figure 3.4: Workflow chart of ELM.

### 3.1.3 Input and Output Layers

As ELM is derived from SVR and DBN, we need to determine the inputs and outputs for training each model. The inputs of SVR model are the initial  $m$  points and last  $m$  points (except the final destination) of the trajectory prefix, which give us a total of  $2m$  coordinates or  $4m$  numerical values. When the prefix of the trajectory contains less than  $2m$  points, overlap the initial and last  $m$  points. When the prefix contains less than  $m$  points, repeat the first or last point. The outputs include the predicted longitudinal value and latitudinal value, which in fact acts as the function of regression.

Fig. 3.5 shows the inputs and outputs of the constructed model of DBN. Firstly, we apply  $k$ -means clustering algorithm to partition trip destinations of the training data into  $n$  clusters, denote the centre of the  $i$ th cluster as  $\mathbf{c}_i$ ,  $1 \leq i \leq n$ . The inputs of DBN model are  $2m$  points of trajectory prefix integrated with time-related embedding vectors  $\mathbf{b}_1$ ,  $\mathbf{b}_2$  and  $\mathbf{b}_3$  aforementioned, which are derived by our proposed CFE technique. The

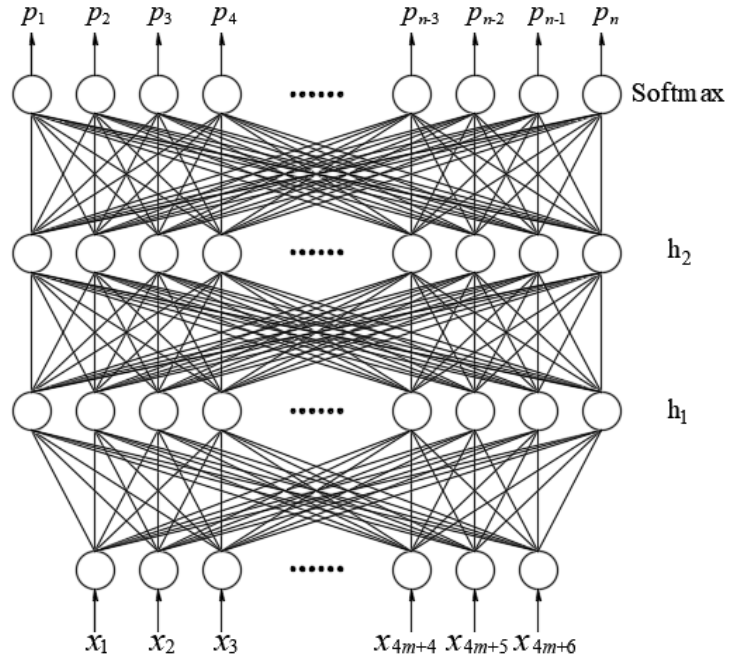


Figure 3.5: Architecture of DBN model.

outputs are the probabilities  $p_i$  corresponding to the  $i$ th cluster, which can be implemented with a Softmax layer on the top layer, as shown in Eq. (3.7).

$$p_i = \frac{\exp(e_i)}{\sum_{j=1}^n \exp(e_j)} \quad (3.7)$$

where  $e_i$  is the  $i$ th activation of the previous layer. Then, the predicted destination can be calculated by Eq. (3.8).

$$\hat{\mathbf{y}} = \sum_{i=1}^n p_i \mathbf{c}_i \quad (3.8)$$

## 3.2 Experiments and Results

### 3.2.1 Datasets and Evaluation Metrics

#### The Porto Dataset

It is a real-world large-scale dataset of taxis in Porto, Portugal (Kaggle 2015). It was collected from 442 taxis running from 1st July 2013 to 30th June 2014. Each observation contains a list of GPS coordinates with longitude and latitude, timestamp and day type (holiday, workday or weekend). The last item of the list represents the destination of this trajectory while the first one corresponds to this trip's pickup location. The time interval between two consecutive GPS coordinates is 15 seconds.

#### The Chengdu Dataset

This dataset is also a real-world dataset collected from more than 14 thousand taxis in the city of Chengdu, China (DataCastle 2016). The period is from 3rd to 30th August 2014. Each observation comprises taxi identity, GPS coordinates, activity (carrying passenger or not) and corresponding timestamp. The GPS data point is recorded with the frequency of every 10 seconds.



### Evaluation Metrics

To evaluate the performance of the proposed models, we use these indices: mean absolute error (MAE) and root mean square error (RMSE). On top of that, for the overall performance evaluation, we define the average MAE (AMAE) and average RMSE (ARMSE). These indexes are defined as

$$d_{ij} = 2r \arcsin\left(\sqrt{\sin^2\left(\frac{\varphi_{ij} - \varphi_i}{2}\right) + \cos \varphi_{ij} \cos \varphi_i \sin^2\left(\frac{\lambda_{ij} - \lambda_i}{2}\right)}\right) \quad (3.9)$$

$$MAE_j = \frac{1}{n} \sum_{i=1}^n d_{ij} \quad (3.10)$$

$$RMSE_j = \sqrt{\frac{1}{n} \sum_{i=1}^n d_{ij}^2} \quad (3.11)$$

$$AMAE = \frac{\sum_{j=1}^m MAE_j}{m} \quad (3.12)$$

$$ARMSE = \frac{\sum_{j=1}^m RMSE_j}{m} \quad (3.13)$$

where  $d_{ij}$  denotes the Haversine distance between the predicted GPS coordinate  $(\varphi_{ij}, \lambda_{ij})$  and real coordinate  $(\varphi_i, \lambda_i)$ .  $r$  is the radius of the earth sphere, we set  $r = 6371\text{km}$  in this research.

### 3.2.2 Experimental Settings

To validate the superiority of our proposed ensemble learning model, we conducted extensive experiments on two real dataset: the Porto dataset and the Chengdu dataset. Our experiments were carried out on a sever with Intel Xeon CPU E5-2680 v2 of 2.8GHz. Some models are subdivided into classification and regression with the postfix of “C” and “R”, respectively. To examine the effects of the proposed ELM within different segments of the whole trajectory, we extract the initial 10% ~ 90% (with increment of 10%) of the whole trajectories for both validation and test sets.

### Data Pre-processing

As the time interval in the Chengdu dataset is only 10 seconds, we convert it into 20 seconds by extracting separated coordinates from the previous trajectory prefix. After dropping some abnormal trajectories, we get a dataset of Porto with totally 1,566,798 trajectories and a dataset of Chengdu with 792,781 trajectories in total. As a practical matter, we always predict destination of current trajectory based on the historical trajectories. Therefore, we arranged the trips by ranking their start time in ascending order, and took the initial 80% and the last 20% as training and test set, respectively. As these datasets are very large, it is quite challenging to train when feeding the whole data. We randomly select 30% candidates for training and 10% of the training data as validation set. Table 3.1 gives the sample sizes of training, test and validation sets.

Table 3.1: Sizes of training, test and validation sets

Dataset	Quantity of Trip		
	Training Set	Test Set	Validation Set
Porto	376031	313360	37603
Chengdu	190268	158556	19027

### Architecture of the Learning Model

There are some parameters  $m$ ,  $\theta_{SVR}$ ,  $\theta_{DBN}$  and  $\lambda$  which should be set before training the models. In our experiment, we take  $m = 5$  for both models. For determining  $\theta_{SVR}$ , after having tried several combinations of parameters on validation set, we choose the radial basis function (RBF) as kernel, and set constant  $C = 100$  and  $\varepsilon = 0.1$  for the insensitive-loss function. For  $\theta_{DBN}$ , the determination of parameters, such as the layer size, neurons in output layer, nodes in each hidden layer, are elaborated later. For  $\lambda$  in ELM, the MAE of different segments in the Porto dataset with SVR and DBN are calculated in Fig. 3.6 (a). It is shown that the overall performance of SVR without CFE

is better, while the DBN-C model with CFE performs best within the initial 30% of the whole trajectory. They perform similarly when the extracted percentage is around 30%, therefore the parameter of  $\lambda$  in our ELM can be set to 0.30. In a similar way, we set  $\lambda = 0.25$  for the dataset of Chengdu.

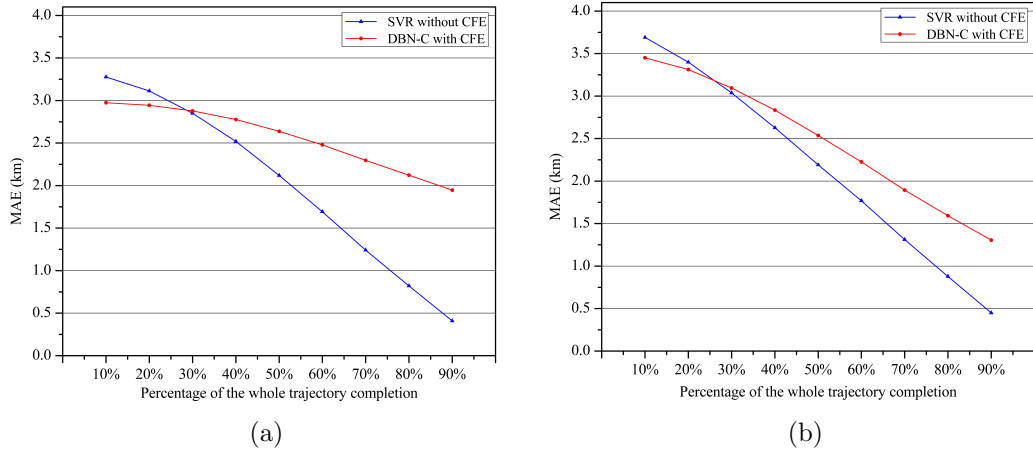


Figure 3.6: Determination of  $\lambda$  with validation sets. (a) on the Porto dataset. (b) on the Chengdu dataset.

### Structure of DBN

To find the best architecture for the DBN model, we test the performance on validation sets with several different architectures, and we choose the structure with the best performance. There are some hyperparameters need to be allocated for DBN. The first is the number of neurons in the output layer. It is chosen from  $\{1800, 2000, 2200, 2400, 2600, 2800\}$  in the Porto dataset, while for the Chengdu dataset, we choose from  $\{600, 800, 1000, 1200, 1400, 1600\}$ . The second is the layer size of network, where layer size from two to seven are chosen to be tested. The third hyperparameter is the number of nodes in each layer, for simplicity, the number of nodes in each layer is set to be the same, and it is chosen from 200 to 1000 for Porto and 50 to 500 for Chengdu.

Table 3.2: Effect of different structures on validation set (on the Porto dataset)

Cluster	AMAE (km)	ARMSE (km)	Node	AMAE (km)	ARMSE (km)	Layer	AMAE (km)	ARMSE (km)
1800	2.60	3.34	200	2.59	3.29	<b>2</b>	<b>2.56</b>	<b>3.29</b>
2000	2.68	3.38	300	2.72	3.47	3	2.84	3.59
2200	2.68	3.41	400	2.60	3.31	4	2.80	3.61
<b>2400</b>	<b>2.59</b>	<b>3.29</b>	<b>500</b>	<b>2.56</b>	<b>3.29</b>	5	2.94	3.69
2600	2.67	3.45	600	2.62	3.33	6	2.77	3.53
2800	2.70	3.42	1000	2.69	3.42	7	3.06	3.84

Table 3.3: Effect of different structures on validation set (on the Chengdu dataset)

Cluster	AMAE (km)	ARMSE (km)	Node	AMAE (km)	ARMSE (km)	Layer	AMAE (km)	ARMSE (km)
600	2.51	3.08	50	2.51	3.06	2	2.50	3.03
<b>800</b>	<b>2.50</b>	<b>3.03</b>	<b>100</b>	<b>2.50</b>	<b>3.03</b>	<b>3</b>	<b>2.47</b>	<b>2.99</b>
1000	2.61	3.17	200	2.55	3.11	4	2.61	3.14
1200	2.68	3.22	300	2.50	3.05	4	3.04	3.68
1400	2.76	3.30	400	2.57	3.13	6	3.37	4.04
1600	3.03	3.54	500	2.60	3.14	7	3.61	4.20

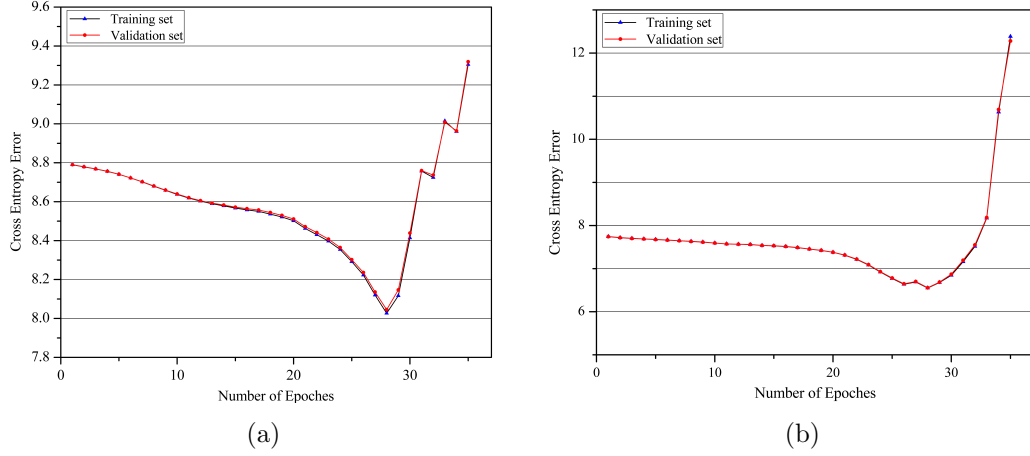


Figure 3.7: Network error with epoch in DBN. (a) on the Porto dataset. (b) on the Chengdu dataset.

Regarding the performances of AMAE and ARMSE on the validation sets, the best structure of DBN for the Porto dataset can be found from Table 3.2: layer size = 2, nodes in layer = 500 and cluster number in output layer = 2400. With Table 3.3, the best structure for Chengdu dataset: layer size = 3, nodes in layer = 100 and cluster number in output layer = 800.

### 3.2.3 Evaluation on the Constructed Classifier

When applying kNN classifier to ELM, we set  $k = 23$  for Porto dataset and  $k = 25$  for Chengdu dataset. Fig. 3.8 shows the accuracy of model for estimating the current segment, which performs as a classifier to allocate the model with the best performance from knowledge base. From Fig. 3.8, we can find that the overall accuracy of the constructed kNN classifier is high, while it performs relatively worse when the percentage of the whole trajectory is around  $\lambda$ . However, this does not make much sense, since the performances of both SVR and DBN-C models are similar when the percentage is near  $\lambda$  (as shwon in Fig. 3.6).

We compare the performance of our proposed ELM with SVR, DBN,

artificial neural network (ANN) [14], kNN and naive Bayes (NB) models. Among these models, some are subdivided into classification and regression models, with the postfix of "C" and "R", respectively. In addition, to evaluate the performance of our proposed CFE technique, each model is also tested with and without CFE technique, respectively.

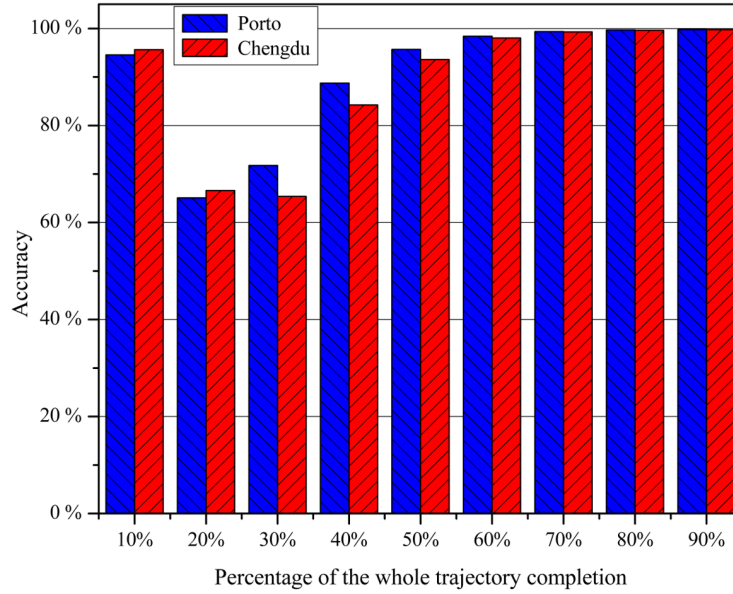


Figure 3.8: Accuracy of kNN classifier for estimating current segment.

### 3.2.4 Comparison with Baseline Methods

Generally, as shown in Table S1 and Table S2 of Supplementary file 1, with the taxi getting closer to the destination, all the prediction models can get higher accuracies. ELM improves the overall performance, and it performs best among all these baselines. Model of SVR outperforms DBN-C when the whole trajectory completion proportion approximates 1, one reason is probably that SVR performs as a regression task to learn the real numerical destinations from training data, while DBN-C is a classification task to learn the probability distributions of destination clusters. However, DBN-

C performs best at the initial proportions, because the pre-trained DBN-C has better generative ability (Gao, Gao, Gao & Wang 2014). Moreover, it can learn invariant features and generate invariant representations from training data, which is insensitive to some transformations and exhibits better classification invariance (Ji, Zhang, Zhang & Wang Aug. 2014, David & Netanyahu 2015). As ELM is derived from SVR mostly, compared with SVR, ELM improves the overall accuracy and it significantly enhances the prediction performance at the initial parts of the whole trajectory. Compared with models of DBN, ANN, kNN and NB, ELM increases the overall performance significantly.

### **3.2.5 Evaluation on the Proposed CFE**

In order to evaluate the effectiveness and efficiency of our proposed CFE technique in features representation, we conduct experiment and compare it with the most commonly used one-hot embedding technique (denotes as One-Hot-E) in our deep learning models. From Table S3 and Table S4 in Supplementary file 1, CFE requires much lower dimensionality for the feature representation compared with the One-Hot-E (26 versus 48 in these experiments), which derives a network with less parameters to be learned from training data. As a result, CFE takes less computational time for training the whole network, especially with complex networks (like deep learning), which has been proved in both the Porto and Chengdu taxi experiments. In addition, CFE employs a fuzzy membership and can well address the similarity issues between the observations, leading to a better prediction performance compared with the similarity-equal One-Hot-E. The proposed ELM is an ensemble model of SVR and DBN, which need to be trained prior to feeding into the segment detection classifier. Table 3.4 demonstrates the time cost of model training and trip’s destination prediction for ELM method.

Table 3.4: Time costs by the our proposed model

Dataset	Training Time		Prediction Time				
	SVR	DBN-C	1 trip	$10^2$ trips	$10^3$ trips	$10^4$ trips	$10^5$ trips
Porto	12.03 h	16.16 h	15.94 s	122.05 s	126.65 s	160.10 s	294.25 s
Chengdu	3.80 h	1.38 h	16.39 s	82.21 s	111.39 s	125.71 s	223.34 s

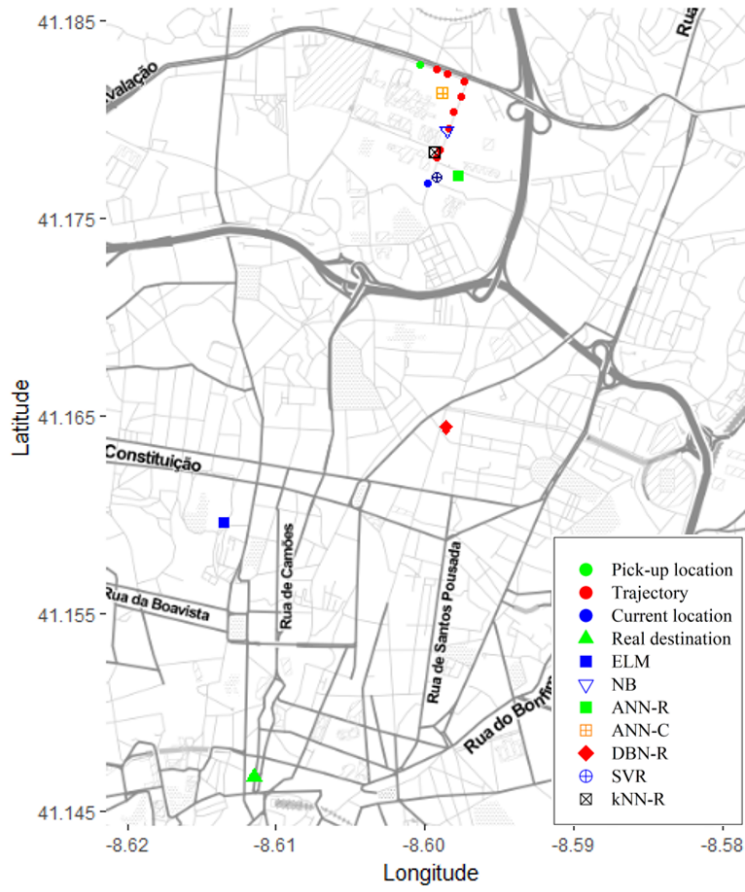


Figure 3.9: Comparison of ELM and other models with a case study (on Porto dataset).



### 3.2.6 Case Study

As presented in Fig. 3.9, assume that a taxi picked up passengers at a location (●), the current location of the taxi is labeled as ●. The real destination of this trip is denoted by ▲. Then, the input of our proposed ELM model is the selected  $2m$  points (the initial  $m$  points counted from pickup location and the last  $m$  points from current location,  $m = 5$  in this experiment) between the pickup location (●) and current location (●). The output is the geographical coordinates of the predicted drop-off location. The predicted destination via ELM, NB, ANN-R, ANN-C, DBN-R, SVR and kNN-R are marked as ■, ▽, ■, □, ◆, ⊕ and ⊠, respectively. The distance of each predicted destination to the real destination is shown in Table 3.5. ELM gets the predicted coordinate at ■, which is the most close to the real drop-off location of this trip. Then various kinds of LBSs based on position ■ can be delivered for taxi riders or driver. Compared with the predicted coordinates derived from other models, the surroundings of the real destination (▲) is more relevant to those around the coordinate predicted via ELM, which is also reasonable and efficient for giving guidelines to taxi riders. While if the predicted destination is too far away from the groundtruth coordinate, it might result in misunderstandings and confusions to users.

Table 3.5: Distances between the real and predicted destinations with different models (unit: km)

ELM	SVR	DBN-R	ANN-R	ANN-C	kNN-R	NB
1.44	3.53	2.25	3.57	3.99	3.66	3.80

## 3.3 Summary

In this chapter, an ensemble learning model is proposed for taxi destinations prediction. In this model, the advantages of SVR and DBN models are incorporated to render a more accurate prediction. A novel data embedding

method named CFE is presented for time-related features embedding in deep learning model. We evaluated the prediction performances and made comparisons with baseline methods of SVR, DBN, ANN, kNN and naive Bayes. Comprehensive experiments on real datasets demonstrate that our ensemble learning model outperforms other baselines in terms of the overall performance. Comparison experiments also confirm that our proposed data embedding method outperforms traditional one-hot embedding in terms of accuracy as well as computational cost.

## Chapter 4

# Offline and Online Detection of Anomalous Patterns from Bus Trajectories for Traffic Insight Analysis

The work in this chapter utilizes deep learning architecture for feature extraction from bus trajectory data sources and develops visualization for both offline and online discoveries of anomalous traffic patterns. We also develop methods for detecting the anomaly locations to provide insights of the corresponding anomalies for understanding the influences caused by the anomaly to the road traffic.

The rest of this chapter is organized as follows. In Section 4.1, we elaborate on the methods of feature extraction and visualization, offline and online algorithms for anomalous patterns detection and insight analysis on anomalies. Section 4.2 presents experimental results and analyses. Section 4.3 makes a summary of this work.

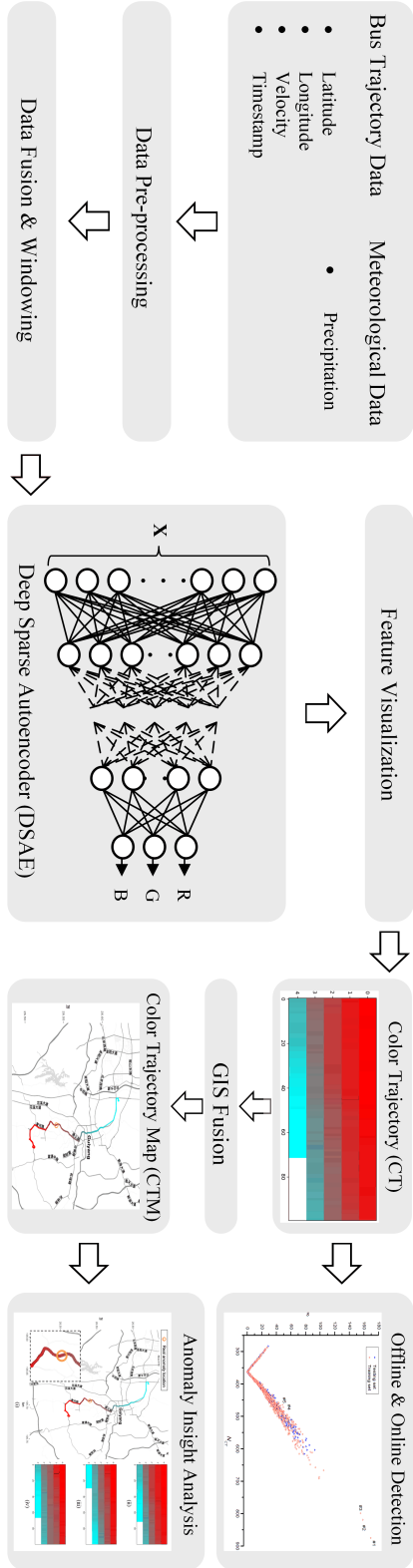


Figure 4.1: The workflow of offline and online detection of anomalous traffic patterns and anomaly insight analysis.

## 4.1 Methods

This section introduces the preliminary definitions, and presents our method for feature extraction and trajectory visualization through deep learning. Then, details of our proposed offline and online methods are described to detect anomalous trajectory and to obtain insights into the anomaly based on the visualized trajectories. A basic workflow of our method is illustrated in Fig. 4.1. An important step of the method is to feed the bus trajectory data and meteorological data into a well trained deep sparse autoencoder (DSAE) to generate the color trajectory (CT), which provides the basis for trajectory visualization, offline and online detection of anomalies. Another key sector is to produce a color trajectory map (CTM) by GIS fusion for anomaly insight analysis.

### 4.1.1 Preliminaries

**Definition 1** *Trajectory:* A trajectory  $\mathbf{T}$  of a moving objective is a set of time-ordered data points,  $\mathbf{T} = (\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_{N-1}, \mathbf{t}_N) \in \mathbb{R}^{D \times N}$ ,  $\mathbf{t}_i = (\varphi_i, \lambda_i, v_i)^T \in \mathbb{R}^3$ , where each data point consists of latitude  $\varphi_i$ , longitude  $\lambda_i$  and velocity  $v_i$  at the  $i$ th timestamp.

**Definition 2** *Class A Anomaly:* Given a trajectory  $\epsilon_i$ , if the extracted spatial feature  $(\tau_i, \tau_k)$  and temporal feature  $N_{CT_i}$  are both very different from the spatial and temporal features of its spatio-temporal neighbors, we termed it as Class A Anomaly.

**Definition 3** *Class B Anomaly:* Given a trajectory  $\epsilon_i$ , if the extracted spatial feature  $(\tau_i, \tau_k)$  is very different from the spatial features of its temporal neighbors, it is defined as Class B Anomaly.

### 4.1.2 Feature Extraction and Trajectory Visualization Using Deep Learning

The method employs a nonlinear dimensionality reduction method (DSAE) to extract hidden features from bus trajectory data to characterize the trajectories for trajectory visualization.

As mentioned in Definition 1, a trajectory is a time series of data points with the same time interval, each data point is typically consisted of latitude  $\varphi$ , longitude  $\lambda$  and velocity  $v$  (unit: km/h). The speed information is popularly available in many existing GPS devices. However, it can also be approximated by algorithm in literature (Feng & Timmermans 2013) in some cases of speed data lack.

Rainfalls, especially heavy rains, can significantly affect traffic flow characteristics and may lead to traffic congestions or even accidents (Jia, Wu & Xu 2017). We integrate the bus trajectory data with local precipitation data  $r$  (unit: mm/h). Thus,  $\mathbf{t}_i$  is updated as  $\mathbf{z}_i$  denoted by

$$\mathbf{z}_i = (\varphi_i, \lambda_i, v_i, r_i)^T \in \mathbb{R}^4 \quad (4.1)$$

and  $\mathbf{T}$  is updated as  $\mathbf{Z}$  denoted by

$$\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_{N-1}, \mathbf{z}_N) \in \mathbb{R}^{(D+1) \times N} \quad (4.2)$$

Data normalization is conducted to normalize the data in each dimension into range  $[-1, 1]$ . For example, the dimension of longitude  $\lambda$  is normalized by Eq. (4.3).

$$\lambda'_i = 2\left(\frac{\lambda_i - \lambda_{min}}{\lambda_{max} - \lambda_{min}}\right) - 1 \quad (4.3)$$

where  $\lambda_{max}$  and  $\lambda_{min}$  are the maximum and minimum values of the longitudinal feature in training set.

Windowing operations is performed as it has been validated that windowing processing could smooth the noise in a relevant study (Liu, Taniguchi, Tanaka, Takenaka & Bando 2017). Suppose a time window size

$\omega$  is set to move  $\mathbf{z}_i$  along the time axis. The windowed data point  $x_i$  and time series  $\mathbf{X}$  are denoted by

$$\mathbf{x}_i = (\varphi_i', \lambda_i', v_i', r_i', \dots, \varphi_{i+\omega-1}', \lambda_{i+\omega-1}', v_{i+\omega-1}', r_{i+\omega-1}')^T \in \mathbb{R}^{4*\omega} \quad (4.4)$$

and

$$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{N_X-1}, \mathbf{x}_{N_X}) \in \mathbb{R}^{(4*\omega) \times N_X} \quad (4.5)$$

where  $N_X = N - \omega + 1$ ,  $\omega$  is an integer and  $0 < \omega < N$ .

$\mathbf{X}$  is then fed into DSAE, which is a deep neural network stacked by many single sparse autoencoders (SSAE). Each SSAE is layer-wise pre-trained before fine-tuning of the whole network. Suppose the visible layer's vector in the  $l$ th SAE is denoted by  $\mathbf{v}^{(l)} \in \mathbb{R}^{D_V \times N_X}$ , then the hidden layer's vector  $\mathbf{h}^{(l)}$  and the reconstruction vector  $\mathbf{r}^{(l)}$  are defined as

$$\mathbf{h}^{(l)} = \tanh(\mathbf{W}_{en}^{(l)} \cdot \mathbf{v}^{(l)} + \mathbf{b}_{en}^{(l)}) \in \mathbb{R}^{D_H^{(l)} \times N_X} \quad (4.6)$$

and

$$\mathbf{r}^{(l)} = \tanh(\mathbf{W}_{de}^{(l)} \cdot \mathbf{h}^{(l)} + \mathbf{b}_{de}^{(l)}) \in \mathbb{R}^{D_R^{(l)} \times N_X} \quad (4.7)$$

where  $\mathbf{W}_{en}^{(l)}$  and  $\mathbf{W}_{de}^{(l)}$  are the weights of the  $l$ th layer of the encoder and decoder, respectively.  $\mathbf{b}_{en}^{(l)}$  and  $\mathbf{b}_{de}^{(l)}$  are the biases of the  $l$ th layer of the encoder and decoder, respectively.

Then, the reconstruction error is calculated by

$$\min L^{(l)} = \frac{1}{2} \|\mathbf{h}^{(l)} - \mathbf{r}^{(l)}\|_2^2 + \alpha \left( \|\mathbf{W}_{en}^{(l)}\|_2^2 + \|\mathbf{W}_{de}^{(l)}\|_2^2 \right) + \beta \sum_{j=1}^{D_H^{(l)}} \text{KL} \left( \rho \parallel \hat{\rho}_j^{(l)} \right) \quad (4.8)$$

and

$$\text{KL} \left( \rho \parallel \hat{\rho}_j^{(l)} \right) = \rho \log \frac{\rho}{\hat{\rho}_j^{(l)}} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho}_j^{(l)}} \quad (4.9)$$

where the L2-norm penalty item is used to prevent over-fitting, and the Kullback-Leibler (KL) divergence is mainly for obtaining a sparse hidden layer to generate more outstanding features.  $\alpha$ ,  $\beta$  and  $\rho$  are the preset

hyperparameters to control the corresponding penalty items, and  $\hat{\rho}_j^{(l)}$  is the average activation of the units in the  $l$ th hidden layer.

A 3-neuron layer is embedded as the output of DSAE to get 3-dimensional hidden features for better visualization, representing the red, green and blue channel in the rgb color space, which is denoted by

$$\mathbf{Y} = (\mathbf{o}_r, \mathbf{o}_g, \mathbf{o}_b)^T \in \mathbb{R}^{3 \times N_Y} \quad (4.10)$$

where  $N_Y = N - \omega + 1$ .

The red channel  $\mathbf{o}_r$  is normalized into range  $[0, 255]$  using Eq. (4.11).

$$\mathbf{R} = \text{Round} \left( \frac{\mathbf{o}_r - \min(\mathbf{o}_r)}{\max(\mathbf{o}_r) - \min(\mathbf{o}_r)} \times 255 \right) \quad (4.11)$$

and similarly for the green channel ( $\mathbf{G}$ ) and blue channel ( $\mathbf{B}$ ).

Then the *color trajectory* (CT) of the trip  $\mathbf{T}$  is denoted by

$$\mathbf{CT} = (\mathbf{R}, \mathbf{G}, \mathbf{B})^T \in \mathbb{R}^{3 \times N_{CT}} \quad (4.12)$$

where  $N_{CT} = N - \omega + 1$ .

### 4.1.3 Offline Anomalous Traffic Patterns Detection

For the  $i$ th complete trajectory, we define  $\boldsymbol{\tau}_i$  as

$$\boldsymbol{\tau}_i = (N_{CT_i}, \mathbf{CT}_i) = \left( N_{CT_i}, (\mathbf{R}_i, \mathbf{G}_i, \mathbf{B}_i)^T \right) \quad (4.13)$$

where  $N_{CT_i} = N_i - \omega + 1$  is a temporal feature that highly depends on the trajectory duration  $N_i$ . A larger  $N_{CT}$  indicates that traffic anomaly might have occurred with higher confidence. However, a trip even with a normally ranged  $N_{CT_i}$  might also be affected by traffic anomalies. Here,  $\boldsymbol{\tau}_i$  is referred to as a *trajectory representation*.

We choose a trajectory representation  $\boldsymbol{\tau}_k$  as *exemplar*. We recommend to choose one with a relatively small  $N_{CT}$ , as it is more unlikely to be anomaly. We denote  $s(\boldsymbol{\tau}_i, \boldsymbol{\tau}_k)$  to represent the similarity between  $\mathbf{CT}_i$  and  $\mathbf{CT}_k$  (i.e., the color trajectory of the exemplar). If  $s(\boldsymbol{\tau}_i, \boldsymbol{\tau}_k)$  is lower, then it is more



similar between  $\mathbf{CT}_i$  and  $\mathbf{CT}_k$ . To compute the similarity, there is a precondition that  $N_{CT_i} = N_{CT_k}$ . If  $N_{CT_k} < N_{CT_i}$ , we append  $N_{CT_i} - N_{CT_k}$  number of points of white color (rgb(255, 255, 255)) to  $\mathbf{CT}_k$  to construct a new trajectory representation  $\tau_j$  to make  $N_{CT_i} = N_{CT_j}$ , while the temporal feature  $N_{CT_k}$  stays the same.

$$\tau_j = (N_{CT_k}, \mathbf{CT}_j) = \left( N_{CT_k}, (\mathbf{R}_j, \mathbf{G}_j, \mathbf{B}_j)^T \right) \quad (4.14)$$

Similarly, if  $N_{CT_k} > N_{CT_i}$ , we do the same processing on  $\mathbf{CT}_i$ , and then get  $\tau_m$ .

$$\tau_m = (N_{CT_i}, \mathbf{CT}_m) = \left( N_{CT_i}, (\mathbf{R}_m, \mathbf{G}_m, \mathbf{B}_m)^T \right) \quad (4.15)$$

Then, the similarity between  $\mathbf{CT}_i$  and  $\mathbf{CT}_k$  can be derived by Eq. (4.16), when  $N_{CT_i} = N_{CT_k}$  or Eq. (4.17), when  $N_{CT_i} \neq N_{CT_k}$ .

$$s(\tau_i, \tau_k) = \sum_{n=1}^{N_{CT_i}} \left( \frac{(\mathbf{R}_i^n - \mathbf{R}_k^n)^2 + (\mathbf{G}_i^n - \mathbf{G}_k^n)^2 + (\mathbf{B}_i^n - \mathbf{B}_k^n)^2}{255^2 + 255^2 + 255^2} \right) \quad (4.16)$$

$$s(\tau_i, \tau_k) = \begin{cases} s(\tau_i, \tau_j) & \text{if } N_{CT_i} > N_{CT_k} \\ s(\tau_k, \tau_m) & \text{if } N_{CT_i} < N_{CT_k} \end{cases} \quad (4.17)$$

Let  $d_{ab}^n = \frac{(\mathbf{R}_a^n - \mathbf{R}_b^n)^2 + (\mathbf{G}_a^n - \mathbf{G}_b^n)^2 + (\mathbf{B}_a^n - \mathbf{B}_b^n)^2}{255^2 + 255^2 + 255^2}$ . Given a small positive threshold  $\varepsilon$ , if the similarity between two color points is smaller than  $\varepsilon$ , we ignore the nuance and redefine the similarity as 0. Therefore, we have Eq. (4.18) in Eq. (4.17).

$$d_{ab}^n = \begin{cases} d_{ab}^n & \text{if } d_{ab}^n \geq \varepsilon \\ 0 & \text{if } d_{ab}^n < \varepsilon \end{cases} \quad (4.18)$$

For the  $i$ th complete trajectory, we have

$$\epsilon_i = (N_{CT_i}, s(\tau_i, \tau_k)) \quad (4.19)$$

where  $s(\tau_i, \tau_k)$  is a spatial feature since it is mainly extracted from buses' GPS spatial positional information, and it can capture the spatial distribution of the moving object.

By mapping all  $\epsilon$  to a two-dimensional space which is referred to as a *spatio-temporal plane* here, we are able to detect those two classes of traffic anomalies defined in Section 4.1.1: *class A anomaly* and *class B anomaly*.

The major differences between *class A anomaly* and *class B anomaly* lies in their neighbors definition and the measurement of similarities between their neighbors. *Class A anomaly* considers both spatial and temporal features to define its neighbors and to measure their similarities. However, if the temporal difference between its neighbors is not significant, there might also be abnormal patterns among them. Therefore, *class B anomaly* reveals this abnormal patterns by addressing the spatial differences from its temporal neighbors. Specifically, if there are several bus trajectories possessed the same or similar temporal features (same or similar trajectory durations), we take them as mutual temporal neighbors. However, if the spatial distribution of one of them is significantly different from the rest, it is understandable that there might be some anomalous events that changed the spatial distribution of this trajectory. Such spatial distribution could be reflected by the spatial feature  $s(\tau_i, \tau_k)$  aforementioned.

Spatio-temporal outliers' co-ordinate points can be detected using our proposed offline anomalous traffic patterns detection (OFF-ATPD) algorithm (Algorithm 4.1), where steps 1 to 11 divide the whole training set into different subsets for class A anomaly detection ( $\epsilon_{train\_C1}$ ) and class B anomaly detection ( $\epsilon_{train\_C2}$ ) by adopting a threshold  $N_C$ . For class B anomaly detection (i.e.,  $\epsilon_i < N_C$ ), we employ the Boxplot rule with a parameter  $\delta$  to identify anomalous observations by aggregating all the spatial features of  $\epsilon_i$  as well as its forward and backward temporal neighbors within  $\eta$  steps (i.e., temporal feature located in  $N_{CT_i} \pm \eta$ ) to form  $S$  (steps 12 to 25). On the other hand, class A anomaly can be detected by computing the Euclidean distance from the nearest spatio-temporal neighbor under a threshold  $r$  (steps 26 to 31).

---

**Algorithm 4.1** OFF-ATPD algorithm

---

**Parameters:**  $N_C, \delta, r, \eta$ .

---

**Input:**  $\epsilon_{train}, \epsilon_i$ . //  $\epsilon_i$  is for test

**Output:**  $C_i$ . // True denotes anomaly

```

1:  $m \leftarrow 0, n \leftarrow 0$ ;
2: for  $\epsilon_j \in \epsilon_{train}$  do
3:    $N_{CT_j} \leftarrow$  Get the temporal feature of  $\epsilon_j$ ;
4:   if  $N_{CT_j} \geq N_C$  then
5:      $m \leftarrow m + 1$ ;
6:      $\epsilon_{train\_C1}(m) \leftarrow \epsilon_j$ ;
7:   else
8:      $n \leftarrow n + 1$ ;
9:      $\epsilon_{train\_C2}(n) \leftarrow \epsilon_j$ ;
10:  end if
11: end for
12: if  $\epsilon_i < N_C$  then
13:    $TN \leftarrow$  Search the forward and backward temporal neighbors of  $\epsilon_i$ 
    from  $\epsilon_{train\_C2}$  within steps of  $\eta$ ;
14:    $S \leftarrow$  Aggregate all the similarities of  $\epsilon_i$  and members in  $TN$ ;
15:    $Q_1 \leftarrow$  Compute the first quartile of  $S$ ;
16:    $Q_3 \leftarrow$  Compute the third quartile of  $S$ ;
17:    $IQR \leftarrow Q_3 - Q_1$ ;
18:    $U \leftarrow Q_3 + \delta * IQR$ ;
19:    $L \leftarrow Q_1 - \delta * IQR$ ;
20:   if  $S(\epsilon_i) > U$  or  $S(\epsilon_i) < L$  then
21:      $C_i \leftarrow \text{True}$ ;
22:   else
23:      $C_i \leftarrow \text{False}$ ;
24:   end if
25: else
26:    $D \leftarrow$  Compute the distance between  $\epsilon_i$  and its nearest spatio-
    temporal neighbor in  $\epsilon_{train\_C1}$ ;
27:   if  $D > r$  then

```

```

28:       $C_i \leftarrow \text{True};$ 
29:  else
30:       $C_i \leftarrow \text{False};$ 
31:  end if
32: end if

```

---

#### 4.1.4 Insight Analysis Using Anomalous Patterns

We combine the trajectory  $\mathbf{T}$  and the color trajectory ( $\mathbf{CT}$ ) in Eq. (4.12) to construct a *color trajectory map* (CTM) through conducting GIS fusion with  $\mathbf{CT}$ . Note that we have  $N_{CT} < N$  after a window size  $\omega$  was introduced in the windowing process. Then we construct a *location vector*  $\mathbf{l}_i$  and a *location matrix*  $\mathbf{L}$ :

$$\mathbf{l}_i = (\varphi_i, \lambda_i)^T \in \mathbb{R}^2 \quad (4.20)$$

and

$$\mathbf{L} = \left( \mathbf{l}_{\lfloor \frac{w-1}{2} \rfloor + 1}, \mathbf{l}_{\lfloor \frac{w-1}{2} \rfloor + 2}, \dots, \mathbf{l}_{\lfloor \frac{w-1}{2} \rfloor + N - w}, \mathbf{l}_{\lfloor \frac{w-1}{2} \rfloor + N - w + 1} \right) \in \mathbb{R}^{2 \times N_L} \quad (4.21)$$

where  $N_L = N_{CT} = N - \omega + 1$ .

We also combine the location matrix  $\mathbf{L}$  with  $\mathbf{CT}$  to generate  $\mathbf{L}'$ :

$$\mathbf{L}' = (\mathbf{L}, \mathbf{CT}) \quad (4.22)$$

For each  $\mathbf{L}'_i$ , map the color with the value of  $(\mathbf{R}_i, \mathbf{G}_i, \mathbf{B}_i)^T$  to coordinate  $(\varphi_i, \lambda_i)^T$  on the GIS map, so as to generate the CTM of a whole trajectory.

$$\mathbf{L}'_i = ((\varphi_i, \lambda_i)^T, (\mathbf{R}_i, \mathbf{G}_i, \mathbf{B}_i)^T) \quad (4.23)$$

The color trajectory (i.e.,  $\mathbf{CT}$  in Eq. (4.12)) and CTM are linked via the conjunct rgb values. By comparing the  $\mathbf{CT}$  of an anomalous trajectory with those non-anomalous trajectories, the most significant difference between them can be found, and the corresponding sector of these colors can be regarded as anomalous. Then, the anomaly occurring location as well as the road influence sector are estimated, by locating the coordinate  $(\varphi_i, \lambda_i)^T$  on the CTM via the anomalous colors  $(\mathbf{R}_i, \mathbf{G}_i, \mathbf{B}_i)^T$  obtained from last step.

#### 4.1.5 Online Detection of Anomalous Traffic Patterns

The proposed OFF-ATPD method in Section 4.1.3 takes the complete bus trajectories as input. It is an offline detection mechanism because the data is ready only after the bus completes the whole trip from the origin place to the terminal stop. In this section, we propose an online anomalous traffic patterns detection (ON-ATPD) algorithm (Algorithm 4.2), which is a substantial extension to the OFF-ATPD algorithm.

By Algorithm 4.2,  $\mathbf{X}_t$  is the real-time input at timestamp  $t$ , derived from Eq. (4.5). Step 1 computes the color trajectory of input  $\mathbf{X}_t$  with DSAE (see details in Section 4.1.2). Step 2 tests whether the bus has arrived at the terminal of the trip or not. Steps 3 and 4 go to the OFF-ATPD algorithm when the bus reaches the terminal stop. While the bus is still on the way to the terminal stop, steps 6 to 13 append or remove segments from the current color trajectory  $\mathbf{CT}_t$  by comparing with the most similar color trajectory from the training set  $\epsilon_{train}$ . Steps 14 and 15 calculate  $\epsilon_{t''}$  with the newly constructed color trajectory  $\mathbf{CT}_{t''}$  and apply the OFF-ATPD algorithm for anomaly detection. Since we have defined the nearest neighbor (the most similar) color trajectory of the real-time  $\mathbf{CT}_t$  (by step 6), there might be a situation that patterns of the nearest neighbor are quite different from the original complete color trajectory. In order to improve the reliability of online anomaly detection, we introduce an integer parameter  $n$  to decide whether all abnormal patterns adjudged from the the previous  $n - 1$  detections and the current detection can yield an anomaly report (steps 16 to 20).

---

##### Algorithm 4.2 ON-ATPD algorithm

---

**Parameters:**  $N_C, \delta, r, \eta, n$ .

**Input:**  $\epsilon_{train}, \mathbf{X}_t, t \geq n$ .

**Output:**  $C_t, t \geq n$ . // True denotes anomaly

- 1:  $\mathbf{CT}_t \leftarrow$  Get the color trajectory of  $\mathbf{X}_t$  with DSAE;
- 2: **if**  $t$  is the end timestamp of the trip **then**
- 3:      $\epsilon_t \leftarrow$  Compute the temporal and spatial features of  $\mathbf{CT}_t$  by Eq. (4.19)  
and go to OFF-ATPD algorithm;

```

4:    $C_t \leftarrow \text{OFF-ATPD}(N_C, \delta, r, \eta, \epsilon_{train}, \epsilon_t);$ 
5: else
6:    $\mathbf{CT}_{t'} \leftarrow$  Get the most similar color trajectory of  $\mathbf{CT}_t$  from  $\epsilon_{train}$ ;
7:    $N_{CT_t} \leftarrow$  Get the temporal feature of  $\mathbf{CT}_t$ ;
8:    $N_{CT_{t'}} \leftarrow$  Get the temporal feature of  $\mathbf{CT}_{t'}$ ;
9:   if  $N_{CT_t} < N_{CT_{t'}}$  then
10:     $\mathbf{CT}_{t''} \leftarrow$  Append  $\mathbf{CT}_t$  with the last  $N_{CT_{t'}} - N_{CT_t}$  color points of
     $\mathbf{CT}_{t'}$ ;
11:   else
12:     $\mathbf{CT}_{t''} \leftarrow$  Remove the last  $N_{CT_t} - N_{CT_{t'}}$  color points of  $\mathbf{CT}_t$ ;
13:   end if
14:    $\epsilon_{t''} \leftarrow$  Compute the temporal and spatial features of  $\mathbf{CT}_{t''}$  and go to
    OFF-ATPD algorithm;
15:    $C_t \leftarrow \text{OFF-ATPD}(N_C, \delta, r, \eta, \epsilon_{train}, \epsilon_{t''});$ 
16:   if  $C_{t-n+1}, \dots, C_{t-1}, C_t$  are all True then
17:     $C_t \leftarrow$  True;
18:   else
19:     $C_t \leftarrow$  False;
20:   end if
21: end if

```

---

## 4.2 Experiments and Results

We have performed comprehensive experiments to answer the following research questions:

**RQ1:** Is OFF-ATPD effective and sensitive to detect all anomalies (i.e., with a high detection rate)?

**RQ2:** Is our developed feature visualization method useful for capturing anomaly locations and traffic impacts with the detected anomalies?

**RQ3:** How does our proposed ON-ATPD perform in real-time traffic anomaly detection?

**RQ4:** How well do our proposed feature extraction deep architecture and

anomaly detection methods perform in comparison with the state-of-the-art methods?

### 4.2.1 Datasets and Evaluation metrics

#### Datasets

We use trajectory datasets from 3 bus routes in Guiyang (China) with a duration of 4 months in the year of 2016. All the data (including the local hourly precipitation data) is officially provided by the Guiyang Open Government Data Platform <sup>1</sup>. The first two datasets are collected on weekends, while the last one is from the off-peak hours (except the morning peak from 06:00 to 09:00 and afternoon peak from 17:00 to 19:30) on weekdays. Each dataset is divided into a training set (the first 3 months) and a test set (the following month). All datasets are naturally unbalanced, since traffic anomalous event rarely occurs along the same bus route. The imbalanced ratios (minority/majority) are 0.025, 0.014 and 0.007 for the test sets of Route 66, Route 50 and Route 18, respectively. Table S1 in Supplementary file 2 provides a detailed description about these datasets.

#### Evaluation Metrics

In the performance evaluation, we use measurements accuracy (Acc), detection rate (DR), false alarm rate (FAR) (Tsai & Lin 2010) and area under the ROC curve (AUC). Criteria of Acc, DR and FAR are defined as follows:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \quad (4.24)$$

$$DR = \frac{TP}{TP + FN} \quad (4.25)$$

$$FAR = \frac{FP}{FP + TN} \quad (4.26)$$

---

<sup>1</sup><http://www.gyopendata.gov.cn/city/index.htm>

- True Positive ( $TP$ ): the number of anomalous trajectory correctly detected as anomaly;
- True Negative ( $TN$ ): the number of non-anomalous trajectory correctly identified as non-anomaly;
- False Positive ( $FP$ ): the number of non-anomalous trajectory incorrectly identified as anomaly;
- False Negative ( $FN$ ): the number of anomalous trajectory falsely identified as non-anomaly.

We also define an index named averaged moving standard deviation (AMSD) to evaluate the concentration of the majority samples (negative samples), which is also a criterion for evaluating the hidden feature extraction architecture. A lower AMSD indicates that those non-anomalies are closer to their neighbors. However, from an overall perspective, a higher AMSD value shows that those non-anomalies are more dissimilar to each other. Method with a higher AMSD might make more false detections, which we should try to avoid in this study. The definition of AMSD can be referred to Eq. (4.27). Firstly, a window size  $\kappa$  for the windowing operation along the horizontal axis  $N_{CT}$  is employed here. Then we compute the sample standard deviation of all the normalized  $s(\boldsymbol{\tau}_{ij}, \boldsymbol{\tau}_k)$  (denoted as  $\hat{s}(\boldsymbol{\tau}_{ij}, \boldsymbol{\tau}_k)$ ) within each  $\kappa$ -sized  $N_{CT}$ . Following this, we get the mean standard deviation of all  $\kappa$ -sized  $N_{CT}$  for AMSD.

$$AMSD = \frac{1}{m} \sum_{i=1}^m \sqrt{\frac{1}{n_i - 1} \sum_{j=1}^{n_i} (\hat{s}(\boldsymbol{\tau}_{ij}, \boldsymbol{\tau}_k) - \bar{s}_i)^2} \quad (4.27)$$

### 4.2.2 Parameters

The parameters are set as:  $(\omega, \alpha, \beta, \rho, \varepsilon) = (10, 10^{-5}, 10^{-4}, 0.05, 0.01)$  for all the bus routes. The window size  $\omega$  cannot be set with either too big or too small value, we choose 10 as suggested by the literature work (Liu, Taniguchi, Tanaka, Takenaka & Bando 2017). We set  $\rho$  with a value near 0



because the centre of each RGB space axis is 0. In addition, the values of  $\alpha$ ,  $\beta$  and  $\varepsilon$  are set empirically, but without using a specific parameters tuning method. Parameter  $\delta$  is a key parameter for detection performance, since too high or too low  $\delta$  will result in a low detection rate or a high false alarm rate (as illustrated by Fig. 4.2). The default value range to determine the upper and lower fences is 1.5 in Boxplot rule. We fine tune the value of  $\delta$  around 1.5. The parameters in both algorithms OFF-ATPD and ON-ATPD are set as the same:  $(N_C, \delta, r, \eta) = (450, 2.0, 50, 2)$  for Bus Route 66,  $(N_C, \delta, r, \eta) = (500, 1.7, 50, 2)$  for Route 50 and  $(N_C, \delta, r, \eta) = (350, 0.9, 40, 2)$  for Route 18, with the understandings and trials from the training set. The Bus Route 18 utilizes a smaller value of  $N_C$  as its route is shorter. The setting of the other parameters in algorithm ON-ATPD is discussed in Subsection 4.2.5. Moreover, we employ a DSAE of four encoding layers with dimensions  $40 \rightarrow 20 \rightarrow 10 \rightarrow 3$  to identify the 3-dimensional hidden features<sup>2</sup>.

### 4.2.3 Offline Detection Results about Anomalous Patterns

The performance comparisons between our proposed OFF-ATPD versus the state-of-the-art baselines are listed in Table 4.3 (note that we have transferred the anomalous observations from the training set to the test set to enlarge the positive sample size for performance evaluation). The proposed OFF-ATPD detects all known anomalies with a high accuracy and a low false alarm rate. The spatio-temporal planes for Bus Routes 66, 50 and 18 are shown in Fig. S1 of Supplementary file 3, where those points distributed along the tick ( $\checkmark$ ) sign exhibit a trend that the  $s$  similarity increases with  $N_{CT}$  when  $N_{CT} > N_{CT_k}$ , while it decreases with  $N_{CT}$  when  $N_{CT} < N_{CT_k}$ . Also in Fig. S1 (a) of Supplementary file 3, anomalies #1 and #2 are categorized as *class A anomalies* as their spatial and temporal features are both far away from their spatio-temporal neighbors, and similarly for anomalies #1, #2, and #3

---

<sup>2</sup>The layer number and neuron number can be changed. However, the network should be a deep learning architecture.

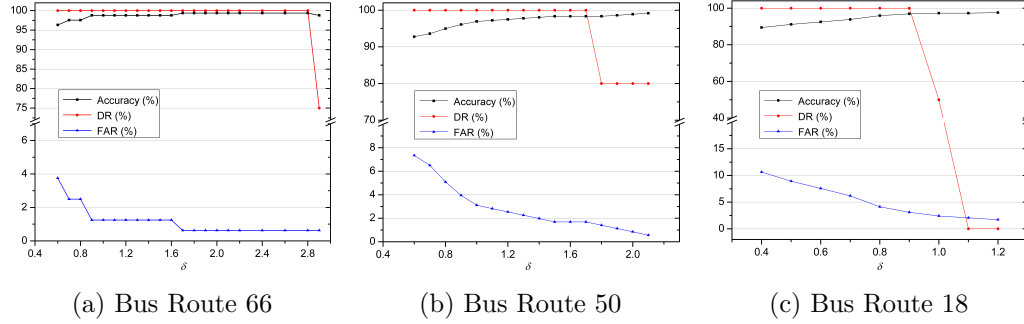


Figure 4.2: Performance comparisons under different settings of parameter  $\delta$ .

in Fig. S1 (b) of Supplementary file 3.

However, anomalies #3, #4 in Fig. S1 (a), #4, #5 in Fig. S1 (b) and #1, #2 in Fig. S1 (c) of Supplementary file 3 were detected as *class B anomalies*, because only their spatial features are far away from their temporal neighbors. In general, class A anomaly has more serious impact on traffics than class B anomaly does, while class B anomaly is more difficult to uncover. Fig. S2 of Supplementary file 3 illustrates the processes of detecting class B anomalies, with steps 14 to 26 in Algorithm 4.1. Fig. 4.2 presents the performance on all the datasets under different settings of parameter  $\delta$ . With the increase of  $\delta$ , a higher accuracy and lower false alarm rates were achieved for all datasets. However, when  $\delta$  rises above a threshold (e.g.,  $\delta > 2.8$  for Bus Route 66), the detection rate decreases, while it remains high when  $\delta$  is located within the threshold.

The detected anomalies shown in Table 4.1 are all coincided with the known traffic anomalous events, which are elaborated as follows:

*Known event 1:* A sedan bumped a car at Shachong East Road in the late afternoon of 14 August 2016, the driver of the sedan escaped after the accident resulting in serious traffic congestion<sup>3</sup>. It was raining at that time and this event only affected services for Bus Route 50.

<sup>3</sup>[http://www.gywb.cn/content/2016-08/16/content\\_5188212.htm](http://www.gywb.cn/content/2016-08/16/content_5188212.htm)

*Known event 2:* A severe car crash (an SUV and a truck) occurred on the West No.2 Ring Road in the morning of 18 September 2016. Two men died on site and one got injured<sup>4</sup>. This event imposed impacts on Bus Route 66 and Route 50 bus services.

*Known event 3:* Two cars crashed on the facilities of a bus station near the Guizhou Cancer Hospital (West Beijing Road) around the noon on 26 November 2016. A pedestrian died<sup>5</sup>. This event affected Bus Route 66 service.

*Known event 4:* An SUV crashed an electric motorcycle on the North Wenchang Avenue in the morning of 14 December 2016. Two riders on the electric motorcycle got injured while trapping under the vehicle<sup>6</sup>. Only Bus Route 18 service was influenced by this crash.

#### 4.2.4 Results about Feature Visualization and Anomaly Insight Analysis

Fig. 4.3 (a), (e) or (i) depicts the CT of a real-world trajectory in Bus Route 66, 50 or 18, respectively. It is evident from Fig. 4.3 (a) that the bus trajectory starts at the color of yellow ■; the color changes gradually and finally gets to blue ■ when the bus is approaching to the destination. The horizontal axis indicates the temporal feature ( $N_{CT}$ , 1 unit equals 10 seconds, each row contains 100 units).

The CTM of anomalous trajectory is obtained by fusion of the color trajectory (CT) with the GIS map (via Eq. (4.22) and Eq. (4.23)). Here we illustrate an anomalous trajectory by taking the anomaly #1 in Bus Route 66 as example. As shown in Fig. 4.4 (a), subfigure (i) is the CTM of #1, and ○ denotes the actual event site. By contrasting the CT of anomaly #1 (i.e., subfigure (ii)) and non-anomalies (i.e., subfigure (iii) and (iv)), we can

---

<sup>4</sup>[http://www.sohu.com/a/114567218\\_398062](http://www.sohu.com/a/114567218_398062)

<sup>5</sup>[https://m.sohu.com/n/474230721/?wscrid=53843\\_3&\\_smuid=](https://m.sohu.com/n/474230721/?wscrid=53843_3&_smuid=BnKG38irJV6gorGDwjyzS0&mv=2)

[BnKG38irJV6gorGDwjyzS0&mv=2](https://m.sohu.com/n/474230721/?wscrid=53843_3&_smuid=BnKG38irJV6gorGDwjyzS0&mv=2)

<sup>6</sup><http://gz.sina.com.cn/news/sh/2016-12-15/detail-ifxytqav9265554.shtml>

Table 4.1: Detected anomalies for each bus route

Bus Route	Anomaly	Service Date	Running Time	Event	Anomaly Category
66	#1	18 Sept. 2016	07:30-09:01	Event 2	Class A anomaly
	#2	18 Sept. 2016	07:00-08:23	Event 2	Class A anomaly
	#3	26 Nov. 2016	12:43-13:55	Event 3	Class B anomaly
	#4	26 Nov. 2016	12:07-13:09	Event 3	Class B anomaly
50	#1	18 Sept. 2016	06:58-09:25	Event 2	Class A anomaly
	#2	18 Sept. 2016	07:22-09:40	Event 2	Class A anomaly
	#3	18 Sept. 2016	07:34-09:48	Event 2	Class A anomaly
	#4	14 Aug. 2016	19:41-21:01	Event 1	Class B anomaly
18	#5	14 Aug. 2016	17:32-18:51	Event 1	Class B anomaly
	#1	14 Dec. 2016	09:31-10:28	Event 4	Class B anomaly
18	#2	14 Dec. 2016	09:50-10:48	Event 4	Class B anomaly

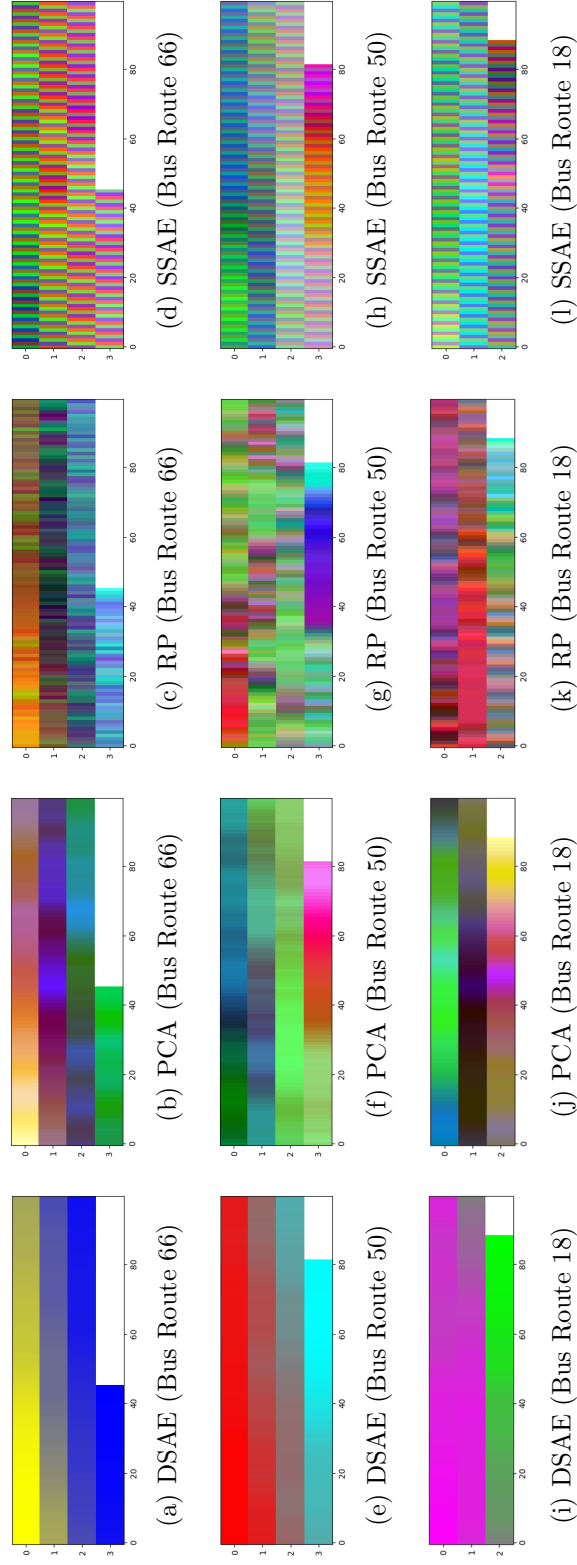
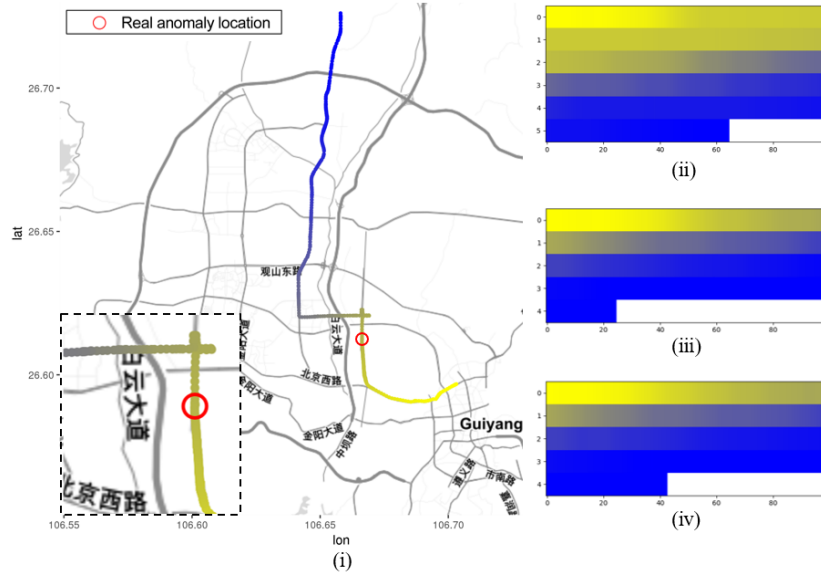
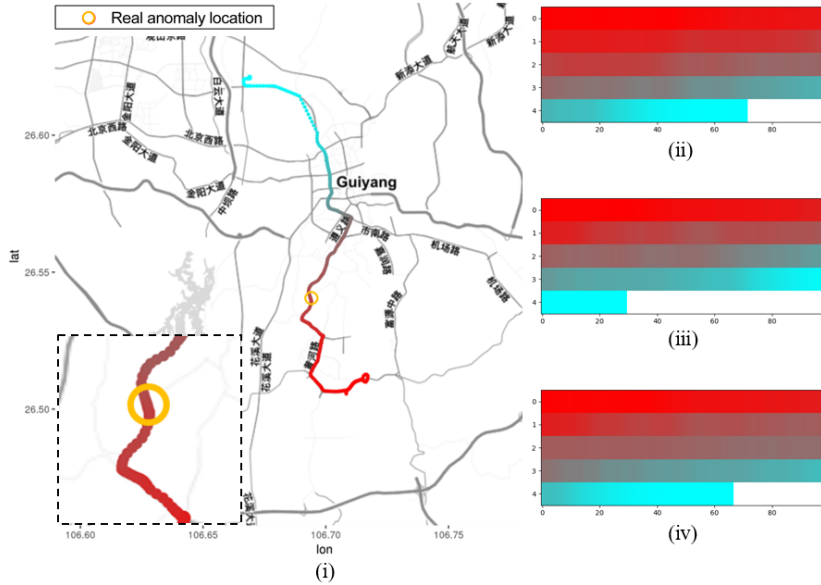


Figure 4.3: Examples of color trajectories generated by DSAE, PCA, RP or SSAE. Each method generates similar visualization patterns on all of these datasets. The CT visualizations generated by our proposed DSAE model are the smoothest.



(a) Anomaly #1 in Bus Route 66



(b) Anomaly #4 in Bus Route 50

Figure 4.4: Insight analyses for anomaly #1 in Bus Route 66 and anomaly #4 in Bus Route 50. (i) CTM of the anomalous trajectory. (ii) CT of the anomalous trajectory. (iii) CT of a non-anomalous trajectory. (iv) CT of another non-anomalous trajectory.

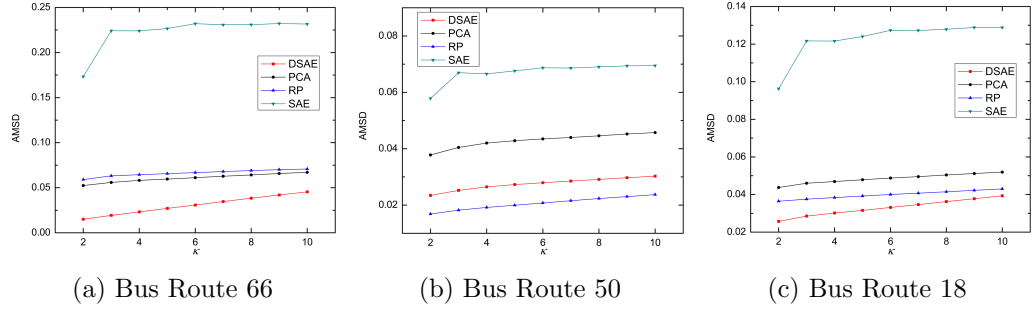


Figure 4.5: Comparison of concentration performance on the training sets.

have an intuitive perspective that the anomaly might have occurred around light yellow ■, because the part with such color is very different from those of the non-anomalies. However, when it proceeds to the color of grey ■, the rest part of CT turns to be similar to those of non-anomalies. It means that the anomaly happened at the locations highlighted between locations ■ and ■ in Fig. 4.4 (a), which is in line with the real location (○) of event 2.

Apart from location detection, our method also provided insights to understand implications of the car crash on the road by highlighting the road section between ■ and ■ (at the left bottom of Fig. 4.4 (a)). Similarly, Fig. 4.4 (b) visually illustrates another example of anomaly #4 in Bus Route 50 that happened between the color of bright red ■ and dark red ■, which also coincides with the real site (○) of event 1.

#### 4.2.5 Online Detection Results about Anomalous Patterns




We conducted online detection simulation experiments for all of the trajectories in test sets. The online anomaly report is carried out every 3 minutes. The parameters  $N_C$ ,  $\delta$ ,  $r$  and  $K$  set for ON-ATPD are the same as those used by OFF-ATPD. Only parameter  $n$  is tested with different values from 1 to 3. Table 4.2 shows the performance of our proposed online detection algorithm. All of the known anomalies are detected correctly. In particular with the increase of  $n$ , higher accuracies and lower false alarm

Table 4.2: Performance of proposed online anomaly detection method (ON-ATPD)

Route	Parameter	Acc (%)	DR (%)	FAR (%)	Time
66	n=1	94.51	100	5.63	3.2s
	n=2	95.12	100	5.00	3.2s
	n=3	<b>97.56</b>	<b>100</b>	<b>2.50</b>	3.2s
50	n=1	91.92	100	8.19	12.1s
	n=2	92.76	100	7.34	12.0s
	n=3	<b>95.26</b>	<b>100</b>	<b>4.80</b>	12.3s
18	n=1	74.40	100	25.77	6.5s
	n=2	76.45	100	23.71	6.5s
	n=3	<b>87.71</b>	<b>100</b>	<b>12.37</b>	6.6s

*Notes:* Time is the mean computational time for one detection. Our experiments were conducted on the server with Intel Xeon Gold 6150 of 2.7GHz.

rates can be achieved on all datasets. On average for each detection, the method needs about 3 seconds of computational time for each detection in Bus Route 66, while needs about 7 and 12 seconds in Bus Routes 18 and 50 detection, respectively.

Fig. 4.6 illustrates some sequential steps of the online detection for anomaly #1 in the Bus Route 66 dataset, where the real-time color trajectories with an interval of 3 minutes are displayed at corresponding timestamp. A detection result of ‘Anomaly’ or ‘No anomaly’ indicates whether there exists any anomaly for the current trajectory. For anomaly #1, the detection system is alarmed around 08:07 AM with an anomaly reporting, when the bus is located at the color . Comparing with the real anomaly location shown in Fig. 4.4 (a), the detected site at color  is quite close to the real anomaly location .



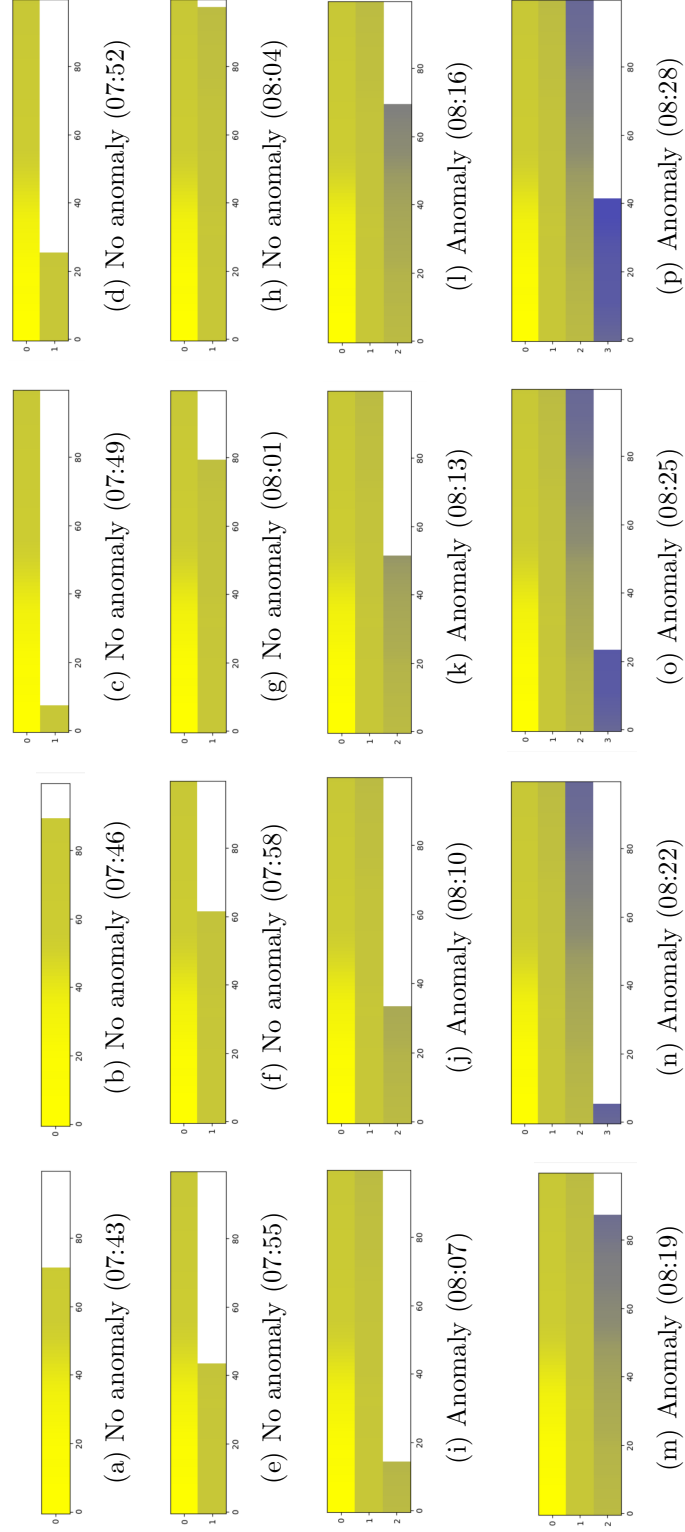



Figure 4.6: An illustrative example of online traffic anomaly detection process in Bus Route 66.

## 4.2.6 Comparison with Baseline Methods

### Feature Extraction and Visualization

Our deep learning-based feature extraction method DSAE is compared with other popular baseline methods including PCA, random projection (RP) and single sparse autoencoder (SSAE) to understand the quality of our color trajectories (CT). From Fig. 4.3, it is apparent that our DSAE-based model can generate the smoothest color distributed trajectories. In Fig. 4.3 (a)(e) and (i), with the trajectory moves on, it gradually changes from one color to another distinct color. While the trajectories by the rest baselines sometimes switch back to the previous color at certain parts of the CT. This conflict will make it difficult for anomaly insight analyses when they are overlapped on the GIS map. Furthermore, on the spatio-temporal planes derived by the above baseline methods, none of them can get better detection performance than the DSAE-based method for all the datasets (Fig. S3 of Supplementary file 3). The distribution of some known anomalies (especially the class B anomalies) yields a similar pattern with that of non-anomalies (#3, #4 in Bus Route 66 with PCA and SSAE, #4, #5 in Bus Route 50 with RP and SSAE, #1, #2 in Bus Route 18 with PCA, RP and SSAE), which makes difficulties to clearly distinguish between anomalies and non-anomalies. Moreover, many of the known non-anomalies are obviously mapped as isolated outlier points (labeled  in Fig. S3 of Supplementary file 3), which do not exhibit the characteristics of the expected patterns.

We also calculated the AMSD values (see Subsection 4.2.1) for all of the non-anomalies, under every window size  $\kappa$  from 2 to 10. DSAE-based model achieved the best performance on the datasets of Bus Route 66 and Route 18, as shown in Fig. 4.5. RP obtained fairly good performance on Bus Route 50; however, its performance in anomaly detection is sensitive as it made false predictions on quite a number of points in Bus Route 50. SSAE performed the worst on all of the datasets.

Table 4.3: Performance comparison on the test sets with the baseline methods

Route	Metric	<b>OFF-ATPD</b>	OneSVM	BiSVM	LSTM	HDBSCAN	kNN
66	Acc (%)	<b>99.39</b>	40.24	98.78	98.17	98.17	98.78
	DR (%)	<b>100</b>	<b>100</b>	50.00	25.00	75.00	50.00
	FAR (%)	0.63	61.25	<b>0</b>	<b>0</b>	1.25	<b>0</b>
	AUC (%)	<b>99.61</b>	75.63	99.53	75.00	97.19	75.00
50	Acc (%)	98.33	52.92	<b>99.44</b>	98.89	<b>99.44</b>	<b>99.44</b>
	DR (%)	<b>100</b>	<b>100</b>	60.00	20.00	60.00	60.00
	FAR (%)	1.69	47.74	0	0	0	0
	AUC (%)	99.66	79.52	<b>100</b>	60.00	97.12	80.00
18	Acc (%)	96.93	48.12	—	—	<b>99.32</b>	—
	DR (%)	<b>100</b>	<b>100</b>	—	—	0	—
	FAR (%)	3.09	52.23	—	—	0	—
	AUC (%)	97.25	75.60	—	—	<b>99.14</b>	—

*Notes:* Supervised learning method BiSVM, LSTM or kNN cannot be applied to Bus Route 18 dataset since there is no positive sample in the training set. AUC is computed by the ‘sklearn’ package in Python.

### Comparison on Anomalous Traffic Patterns Detected by Our Offline Approach

We compare the anomaly detection performance by our offline detection approach (OFF-ATPD) with those by the commonly used methods in outlier/anomaly detection (Chandola, Banerjee & Kumar 2009), including classification-based methods (one-class SVM (OneSVM) (Li et al. 2003, Wang et al. 2004), binary SVM (BiSVM) and LSTM network), a clustering-based method (HDBSCAN clustering (Campello, Moulavi & Sander 2013)) and a nearest-neighbor-based method (kNN). The same features extracted via DSAE are used for these baseline methods. Our approach is implemented by Python and Tensorflow, the code of our algorithms is publicly available in GitHub repositories<sup>7</sup>. OneSVM and BiSVM use the ‘e1071’ package in R. LSTM network is implemented by the ‘rnn’ package in R. While baselines of HDBSCAN and kNN use the R packages of ‘dbscan’ and ‘FNN’, respectively.

<sup>7</sup><https://github.com/Xiaocai-Zhang/Anomalous-Traffic-Patterns-Detection>

The output probabilities of our approach to calculate AUC are linearly scaled by the similarity (i.e.,  $S(\epsilon_i)$  in Algorithm 4.1). While the output probabilities via OneSVM and BiSVM are estimated by Platt scaling (Lin, Lin & Weng 2007). The performances are shown in Table 4.3. Because there is no positive sample in the training set of Bus Route 18, the supervised learning methods of BiSVM, LSTM and kNN are not applicable. Overall, OFF-ATPD achieved better performances with high accuracies, the 100% detection rates, low false alarm rates and high AUC scores on all of these datasets. OneSVM is also a competitive method that detected all anomalies correctly; however, its high false alarm rates (61.25%, 47.74% and 52.23%) make it less efficient. BiSVM and HDBSCAN also demonstrated low false alarm rates and high AUC scores; nevertheless, they are unable to identify all the anomalies accurately. None of the rest machine learning baseline methods could detect all of the anomalies correctly. One reason is probably that the real-world datasets for traffic anomaly detection as we utilized in this experiment are extremely imbalanced. Machine learning on imbalanced datasets might produce unsatisfactory classifiers (Provost 2000, Zhu, Lin & Liu 2020). Instead of taking machine learning ideas for pattern recognition, our developed algorithm explores the ideas of spatio-temporal neighborhood and Boxplot rules to identify anomalous traffic patterns in class A task and in class B task, respectively. Because these anomalous patterns have distinct spatial and temporal characteristics, our approach can achieve better performance on imbalanced data than the baseline machine learning approaches.

### 4.3 Summary

The work in this chapter mainly consists of four parts. First, deep learning-based method is proposed to extract novel features from bus trajectory data, and the method can make good visualization of this features as well. Second, we have termed *class A anomaly* and *class B anomaly* to better address

the discrepancy issues between these diversified anomalous patterns. Offline detection algorithm is designed by using the Boxplot rule or the nearest neighborhood for detecting different classes of anomalous patterns. Third, we fused the visualized color trajectories with GIS map to generate a color trajectory map, and developed methods that are able to conduct insights analysis on the locations of anomalies as well as on the traffic influences to the road. Last, we also developed an online detection method extending from the offline method for a real-time detection of anomalous traffic patterns. Extensive experiments on three real-world datasets confirmed the effectiveness and superiority of our deep feature extraction method, offline and online detection methods, and anomaly insight analysis method.

Infrastructure plans for some cities have adopted the ‘Bus Lane’ strategy for some major roads during certain periods to improve the efficiency of bus services. In that case, our approach may not be efficient to detect the incident-based anomaly, as the situation that some incidents affecting other vehicles on the road might not affect buses. However, from the perspective of bus service operation or management, that situation does not affect the decision making, since those anomalies that impose little impact on bus service will not be taken into account.

## Chapter 5

# Differential Evolution based LSTM Recurrent Neural Network for Traffic Flow Prediction

Deep learning has drawn large amount of attention and has made great progresses in traffic flow prediction. State-of-the-art deep learning algorithms like DNN (Qu et al. 2019), SAE (Lv et al. 2014), DBN (Huang et al. 2014, Huang et al. 2013) and LSTM (Tian & Pan 2015) have achieved much better performances in traffic flow prediction than the traditional parametric methods or other machine learning methods. However, global optimization of the network's hyperparameters is still a tough problem in deep learning despite of remarkable improvements. The aforementioned deep learning models in traffic flow prediction or even in other relevant domains (Tang, Liu, Zou, Zhang & Wang 2017, Zhang, Zhao, Zheng & Li 2019, Zhang, Liu, Zheng, Zhao, Li & Liu 2018) all employ a grid search strategy or a random search strategy for hyperparameters tuning. Grid search considers all of the possible combinations of the hyperparameters with specified grid gaps. Such an exhaustive strategy is computationally expensive in the case

of many hyperparameters or large numbers of training samples, which has led to poor performance in practice (Bergstra & Bengio 2012). On the other hand, random search scans over a low-dimensional subspace of all possible combinations with the notion that not all hyperparameters are equally important. Previous study (Bergstra & Bengio 2012) has demonstrated that random search could obtain satisfactory models in most cases while with low computational cost.

We propose to use a differential evolution algorithm (Storn & Price 1997) for deep learning hyperparameters optimization and in particular to obtain optimal hyperparameters of LSTM recurrent neural network (RNN). LSTM is specially capable of learning long-term interdependencies (Liu, Shahroudy, Xu, Kot & Wang 2017) to overcome the gradient vanishing problem in RNN. This advantage makes LSTM effective and efficient in processing sequence data including traffic flow time series data. The differential evolution algorithm is good at optimizing the hyperparameters in LSTM to achieve high accuracy and robustness for traffic flow prediction. The LSTM network of the optimal hyperparameters is then trained to learn important sequential traffic flow features. To accelerate the convergence of the differential evolution algorithm, we design parallel computing and early stopping programs. Our new deep learning method is termed PDE-LSTM, standing for parallel-differential-evolution-based LSTM for traffic flow prediction. This is the first work that employs evolutionary algorithms to optimize the deep learning architecture and hyperparameters in the problem of traffic flow prediction.

The rest of this chapter is structured as follows. Section 5.1 presents details of PDE-LSTM. Section 5.2 describes the datasets collected by Dublin and San Francisco, reports and analyzes the results of the newly proposed PDE-LSTM. Section 5.3 summarizes this chapter.

## 5.1 Methods

The workflow of our proposed model for traffic flow prediction is illustrated by Fig. 5.1. The workflow contains 3 main components: data processing, PDE-LSTM and prediction. First, the preprocessed dataset is split into training data, validation data and test data. Second, PDE-LSTM is performed on the training and validation data to optimize the hyperparameters of LSTM. Third, the traffic flow for a near future time interval is predicted by the optimized LSTM model. In this section, we give a brief introduction to the differential evolution algorithm. Then, details of our proposed traffic flow prediction model, including parallel computing and early stopping steps, are described. Finally, an example is provided to illustrate the steps for a better understanding of the details.

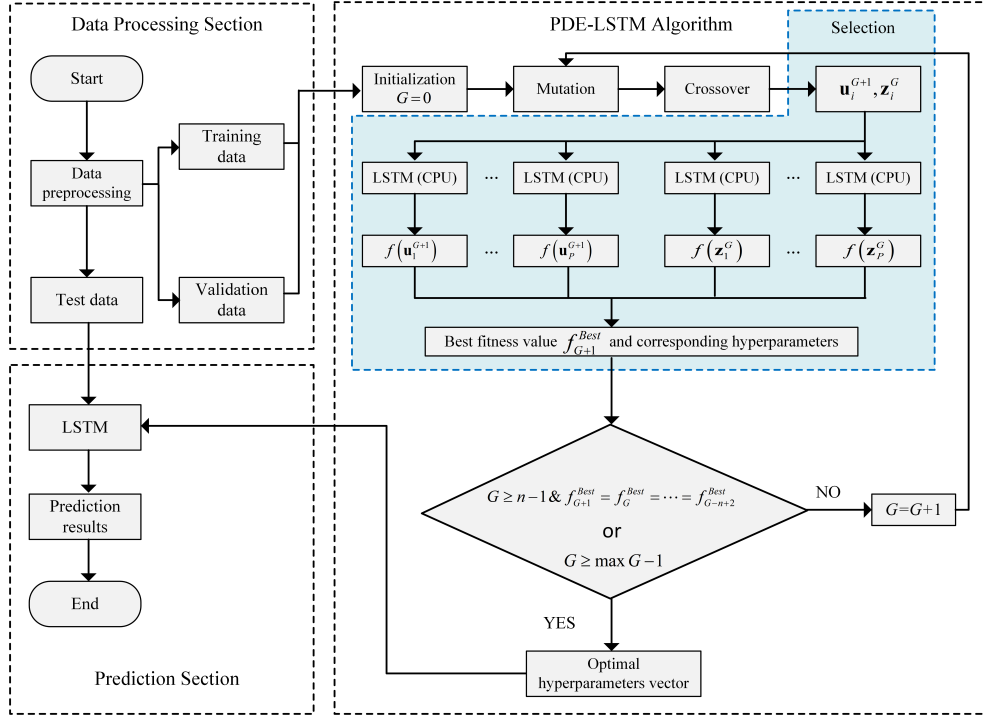


Figure 5.1: The workflow of PDE-LSTM in traffic flow prediction.



### 5.1.1 Differential Evolution

Differential evolution (DE) is proposed by Storn and Price (Storn & Price 1997), which is an efficient population-based stochastic search technique to solve optimization problems (Wu, Shen, Li, Chen, Lin & Suganthan 2018). A standard DE algorithm consists of 4 steps: initialization, mutation, crossover and selection.

#### Initialization

Suppose there are  $D$  parameters to be optimized in a problem. For the  $j$ th parameter,  $j = 1, 2, \dots, D$ , let its corresponding range be  $[b_{L_j}, b_{U_j}]$ , where the subscripts  $L$  and  $U$  denote the lower and upper bounds. The initialized population with a size  $P$  is generated by Eq. (5.1).

$$\mathbf{z}_{ij}^0 = b_{L_j} + rand(0, 1) * (b_{U_j} - b_{L_j}) \quad (5.1)$$

where  $i = 1, 2, \dots, P$ ;  $rand(0, 1)$  is the function to generate a random number valued between 0 and 1 with a uniform probability distribution.

#### Mutation

The purpose of mutation is to add a scaled vector difference between two randomly sampled individuals to a third individual vector (Hamza, Essam & Sarker 2015). The mutation is conducted using Eq. (5.2) and Eq. (5.3).

$$\mathbf{v}_{ij}^{G+1} = \mathbf{z}_{kj}^G + F_i^G * (\mathbf{z}_{lj}^G - \mathbf{z}_{mj}^G) \quad (5.2)$$

and

$$F_i^G = random(F_L, F_U) \quad (5.3)$$

where  $1 \leq i \neq k \neq l \neq m \leq P$ ;  $F_i^G$  is the mutation factor produced randomly from the uniform distribution on the interval  $[F_L, F_U]$ ; superscript  $G$  denotes the  $G$ th generation of the algorithm.

### Crossover

The step of crossover is to cross the initialized and mutated vectors to generate new trail vectors. The DE algorithm crosses each vector using Eq. (5.4) and Eq. (5.5).

$$\mathbf{u}_{ij}^{G+1} = \begin{cases} \mathbf{v}_{ij}^{G+1}, & \text{if } c_{ij}^G \geq C_r \\ \mathbf{z}_{ij}^G, & \text{if } c_{ij}^G < C_r \end{cases} \quad (5.4)$$

and

$$c_{ij}^G = \text{rand}(0, 1) \quad (5.5)$$

where  $\mathbf{u}_{ij}^{G+1}$  is the new trail vector after crossover;  $C_r \in [0, 1]$  is the crossover factor;  $\text{rand}(0, 1)$  generates a random value uniformly distributed on the interval between 0 and 1.

### Selection

The selection operation aims to select the best genes for offsprings. The selection is carried out by comparing the fitness values of the trail vector  $\mathbf{u}_i^{G+1}$  and the target vector  $\mathbf{z}_i^G$ . If the fitness value of trail vector  $\mathbf{u}_i^{G+1}$  is better than that of target vector  $\mathbf{z}_i^G$ , then replace the target vector with trail vector  $\mathbf{u}_i^{G+1}$  for the offspring; otherwise keep target vector  $\mathbf{z}_i^G$  to the next generation until the algorithmic terminal condition is met.

$$\mathbf{z}_i^{G+1} = \begin{cases} \mathbf{u}_i^{G+1}, & \text{if } f(\mathbf{u}_i^{G+1}) \leq f(\mathbf{z}_i^G) \\ \mathbf{z}_i^G, & \text{if } f(\mathbf{u}_i^{G+1}) > f(\mathbf{z}_i^G) \end{cases} \quad (5.6)$$

## 5.1.2 Differential Evolution based LSTM under Parallel Computing

Suppose at time  $t + m$ , we have the historical traffic flow at time intervals  $(t + 1, t + 2, \dots, t + m)$ , the task is to predict the traffic low at time interval  $t + m + 1$ . The input  $\mathbf{X}$  of the model can be represented by Eq. (5.7) and Eq. (5.8).

$$\mathbf{x}_j = (x_{j1}, x_{j2}, \dots, x_{jm})^T \in \mathbb{R}^m \quad (5.7)$$

and

$$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{N-1}, \mathbf{x}_N) \in \mathbb{R}^{m \times N} \quad (5.8)$$

where  $m$  is the sequence length and  $N$  is the number of observations.

The  $\mathbf{yz}_i^G$  denotes the output via the LSTM model under the hyperparameters from the  $i$ th target vector at the  $G$ th iteration  $\mathbf{z}_i^G$ . Similarly,  $\mathbf{yu}_i^{G+1}$  is the corresponding output with hyperparameters derived from the  $i$ th trail vector  $\mathbf{u}_i^{G+1}$ .

$$\mathbf{yz}_i^G = g_{LSTM}(\mathbf{X}, \mathbf{z}_i^G) \quad (5.9)$$

$$\mathbf{yz}_i^G = (yz_{i1}^G, yz_{i2}^G, \dots, yz_{i(N-1)}^G, yz_{iN}^G) \in \mathbb{R}^N \quad (5.10)$$

$$\mathbf{yu}_i^{G+1} = g_{LSTM}(\mathbf{X}, \mathbf{u}_i^{G+1}) \quad (5.11)$$

$$\mathbf{yu}_i^{G+1} = (yu_{i1}^{G+1}, yu_{i2}^{G+1}, \dots, yu_{i(N-1)}^{G+1}, yu_{iN}^{G+1}) \in \mathbb{R}^N \quad (5.12)$$

where  $\mathbf{X}$  stands for the input data;  $g_{LSTM}$  represents the LSTM model with the hyperparameters vector  $\mathbf{z}$  or  $\mathbf{u}$ ;  $\mathbf{yz}$  and  $\mathbf{yu}$  denote the corresponding outputs. We use mini-batch stochastic gradient decent (SGD) together with the RMSProp (Tieleman & Hinton 2012) optimizer to train the LSTM model.

Suppose the groundtruth of traffic flow is denoted by Eq. (5.13).

$$\hat{\mathbf{y}} = (\hat{y}_1, \hat{y}_2, \dots, \hat{y}_{N-1}, \hat{y}_N) \in \mathbb{R}^N \quad (5.13)$$

We employ the criterion of mean absolute percentage error (MAPE) for measuring the fitness values. The fitness values of target vector and trail vector are defined by Eq. (5.14) and (5.15), respectively.

$$f(\mathbf{z}_i^G) = \frac{1}{N} \sum_{j=1}^N \frac{|yz_{ij}^G - \hat{y}_j|}{|\hat{y}_j|} \cdot 100\% \quad (5.14)$$

and

$$f(\mathbf{u}_i^{G+1}) = \frac{1}{N} \sum_{j=1}^N \frac{|yu_{ij}^{G+1} - \hat{y}_j|}{|\hat{y}_j|} \cdot 100\% \quad (5.15)$$

Since training a deep learning model is a time-consuming process, we introduce parallel computing for training the LSTM models and calculating the fitness values of target and trail vectors, as illustrated by the selection procedure in Fig. 5.1. Then, the offsprings  $\mathbf{z}_i^{G+1}$  could be determined by Eq. (5.6). The best fitness value of this generation is defined by Eq. (5.16).

$$f_{G+1}^{Best} = \min f(\mathbf{z}_i^{G+1}) \quad (5.16)$$

In order to improve the time efficacy of hyperparameter optimization process, we propose to integrate a criterion of early stopping into our algorithm. Firstly, a parameter  $n$  is defined, if the best fitness of the  $G$ th generation (i.e.,  $f_{G+1}^{Best}$ ) is the same with the best fitness values of the previous  $n - 1$  generations, the early stopping mechanism will be triggered, as shown by Eq. (5.17).

$$f_{G+1}^{Best} = f_G^{Best} = \dots = f_{G-n+2}^{Best} \quad (5.17)$$

where  $G \geq n - 1$  and  $G \leq \max G - 1$ .

The global optimal hyperparameters vector  $\mathbf{z}^*$  are determined using Eq. (5.18).

$$\mathbf{z}^* = \arg \min_{\mathbf{z} \in \mathbb{R}^D} f(\mathbf{z}_i^{G^*+1}) \quad (5.18)$$

where  $G^*$  denotes the generation that the terminal condition is applied.

### 5.1.3 PDE-LSTM: An Illustrative Example

A simplified example of PDE-LSTM is illustrated at Table 5.1 to Table 5.6. Suppose there are 5 hyperparameters for optimization: sequence length (SL), hidden unit (HU), maximum epoch (ME), batch size (BS) and learning rate (LR); and assume they are in the ranges  $[1, 50]$ ,  $[1, 75]$ ,  $[10, 1000]$ ,  $[1, 500]$  and  $[0.001, 0.1]$ , respectively. If the population size is set as  $P = 4$ , then the initialized population is the numbers shown in Table 5.1.

Table 5.1: Population with the step of initialization

SL	HU	ME	BS	LR
15.62	31.56	286.48	111.46	0.0902
45.55	3.62	967.34	332.06	0.0611
36.47	31.67	450.11	457.8	0.0077
22.10	45.11	786.58	144.83	0.0442

For the step of mutation, we assume  $F_L = 0.5$  and  $F_U = 1$  to generate the mutation factor. Table 5.2 lists the mutated population.

Table 5.2: Population with the step of mutation

SL	HU	ME	BS	LR
16.80	61.46	485.20	218.07	0.0130
11.21	61.62	806.02	12.06	0.0471
27.19	44.97	494.86	332.16	0.0132
16.75	52.81	811.18	73.01	0.0472

In the step of crossover, if the crossover factor is set as  $C_r = 0.7$ , then the population can be updated by Eq. (5.4) and (5.5), as listed in Table 5.3.

Table 5.3: Population with the step of crossover

SL	HU	ME	BS	LR
15.62	61.46	286.48	111.46	0.0902
45.55	3.62	967.34	12.06	0.0611
27.19	31.67	450.11	457.76	0.0132
22.10	45.11	786.58	73.01	0.0472

For the selection step, all the hyperparameters' values in the population are rounded to integers, and parallel computing is employed for the parallel training of LSTM network with the corresponding hyperparameters vector. The fitness values (MAPE) of the target vectors and trail vectors are shown in Table 5.4 and 5.5, respectively.

Table 5.4: Fitness values of initialized population (target vectors)

SL	HU	ME	BS	LR	Fitness (MAPE)
16	32	286	111	0.0902	6.95%
46	4	967	332	0.0611	6.68%
36	32	450	458	0.0077	6.27%
22	45	787	145	0.0442	6.53%

Table 5.5: Fitness values of crossovered population (trail vectors)

SL	HU	ME	BS	LR	Fitness (MAPE)
16	61	286	111	0.0902	6.63%
46	4	967	12	0.0611	6.99%
27	32	450	458	0.0132	6.41%
22	45	787	73	0.0472	6.72%

Table 5.6 shows the rounded numbers of offspring population after the step of selection. If the algorithm terminates at this iteration, then the vector  $\mathbf{z}^* = (36, 32, 450, 458, 0.0077)^T$  with the minimum fitness are regarded as the optimal hyperparameters. If not, the initialized population in step 1 will be updated by the offspring for the next iteration.

Table 5.6: Offspring population with the step of selection

SL	HU	ME	BS	LR	Fitness (MAPE)
16	61	286	111	0.0902	6.63%
46	4	967	332	0.0611	6.68%
<b>36</b>	<b>32</b>	<b>450</b>	<b>458</b>	<b>0.0077</b>	<b>6.27%</b>
22	45	787	145	0.0442	6.53%

## 5.2 Experiments and Results

### 5.2.1 Datasets and Evaluation Metrics

#### Datasets

The proposed method was tested on three real-world road traffic flow datasets. Two of them are about the traffic flow of two roads in Dublin, Ireland, and the third one is about a main road in San Francisco, USA. We choose three roads of dense traffic flow for study because they have been given much attentions in both traffic management and research (Huang et al. 2014). The two Irish datasets were download from the official traffic data site maintained by Transport Infrastructure Ireland (TII) (Transport Infrastructure Ireland 2019). The San Francisco dataset was download from the Caltrans Performance Measurement System (PeMS) database (PeMS 2019) maintained by the California Department of Transportation (Caltrans). All the flow data were collected at a 15-minute interval continuously spanning one year from April 1 2018 to March 31 2019, using the average of all the inductive loop detectors in the corresponding road. The 15-min format of data can be transformed into a 30-min, 45-min or 60-min interval format for different tasks of prediction.

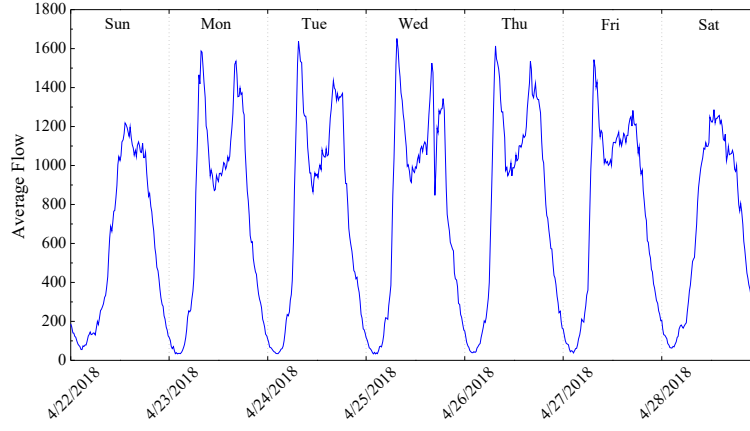
The dataset for each road is divided into a training set, a validation set and a test set. In particular, the flow data of the first 9 months is reserved as the training set, the 10th month’s data for validation, and the last two months’ for test. Table 5.7 provides more details about these datasets. The average traffic flow of 15 minutes interval during a typical week is depicted in Fig. 5.2. All the data and code are publicly available in GitHub repositories<sup>8</sup>.

#### Evaluation Metrics

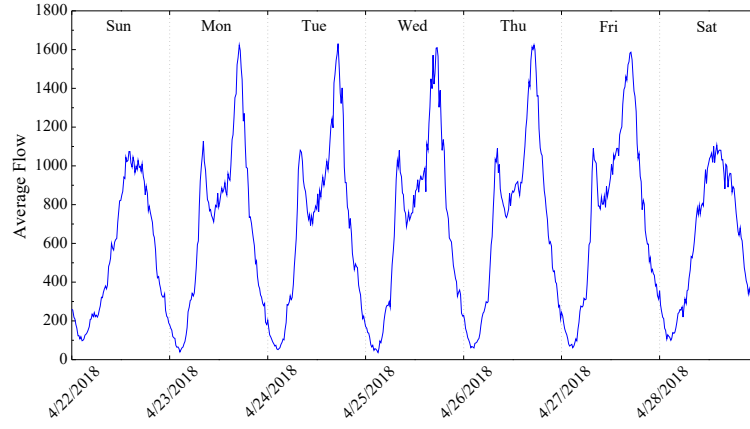
The prediction performance of the proposed method and existing methods are evaluated using three metrics: mean absolute error (MAE), root mean

---

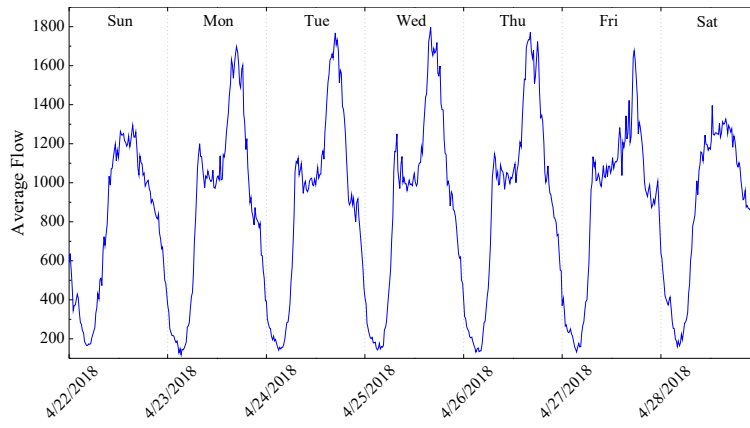
<sup>8</sup>[https://github.com/Xiaocai-Zhang/Traffic\\_flow\\_prediction\\_based\\_DE-LSTM](https://github.com/Xiaocai-Zhang/Traffic_flow_prediction_based_DE-LSTM)



(a) M50-N (15-min)



(b) M1-N (15-min)



(c) I280-S (15-min)

Figure 5.2: Average traffic flow of the M50-N, M1-N and I280-S roads during a typical week.



Table 5.7: Dataset description

Dataset	Road	Segment	Direction	Detector No
M50-N	M50	Balinteer to Finglas	Northbound	8
M1-N	M1	Airport to Swords	Northbound	2
I280-S	I280	Bernal Heights to Ingleside	Southbound	7

square error (RMSE) and mean absolute percentage error (MAPE). They are defined as

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (5.19)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (5.20)$$

$$MAPE = \frac{1}{n} \sum_{i=1}^n \frac{|y_i - \hat{y}_i|}{\hat{y}_i} \cdot 100\% \quad (5.21)$$

where  $y_i$  denotes the  $i$ th predicted traffic flow value, and  $\hat{y}_i$  indicates the corresponding groundtruth flow value.

## 5.2.2 Hyperparameters

For a deep learning model, there are two categories of hyperparameters: model hyperparameters and optimizer hyperparameters. In this study, we choose to use 5 hyperparameters that have heavy impact on the performance of the LSTM regression model. These hyperparameters are: sequence length, hidden unit, maximum epoch, batch size and learning rate. Sequence length and hidden unit are model hyperparameters applicable to determine the structure of LSTM network. The maximum epoch, batch size and learning rate are optimizer hyperparameters that have effects on the training process. More details of these hyperparameters include:

- Sequence length: it refers to the input length of LSTM network, corresponding to  $m$  in Eq. (5.7). An appropriate input length can

capture the most critical information of the data while excluding unnecessary input data;

- Hidden unit: it stands for the number of units in the hidden layer of LSTM network. An over-set number of hidden units may significantly increase the time cost for training and sometimes the process may fail to converge (Ling 1995);
- Maximum epoch: it refers to the maximum number of training epochs that completely pass through the training dataset. A small maximum epoch might result in underfitting, whereas a larger maximum epoch takes more time for training. While training the LSTM network with a maximum setting of epoch, we choose the best model at the epoch that the lowest MAPE is witnessed on the validation set;
- Batch size: a large batch size can speed up the network training process. However, a large batch size requires a huge memory. On the other hand, a small batch size may cause the process difficult to converge (Dai & Zhu 2018);
- Learning rate: the learning rate is an important hyperparameter for SGD algorithm. It controls how fast the learning model is adapt to the problem. Too large learning rate can lead the model to a suboptimal solution, while a learning with too small value may make the training process become permanently stuck.

### **5.2.3 Parameter Settings**

#### **Hyperparameter Ranges**

As required, we have 5 hyperparameters for optimization (i.e,  $D = 5$ ). In order to enhance the practical applicability of our proposed method in practice, we avoid using parameter tuning technique because it is a time-consuming procedure, especially for tuning a deep learning model on large-scale data. Instead, we assign higher upper bounds to all hyperparameters

Table 5.8: Lower and upper bounds of each hyperparameter

Sequence Length		Hidden Unit		Maximum Epoch		Batch Size		Learning Rate	
$b_{L_1}$	$b_{U_1}$	$b_{L_2}$	$b_{U_2}$	$b_{L_3}$	$b_{U_3}$	$b_{L_4}$	$b_{U_4}$	$b_{L_5}$	$b_{U_5}$
1	50	1	75	10	1000	10	500	0.001	0.1

to test the searching ability of DE algorithm. Thus, the ranges of “sequence length” and “hidden unit” are all set as  $[1, 50]$  and  $[1, 75]$ , respectively. The bounds of the “maximum epoch” are all set as  $[10, 1000]$ . All “batch size” are generated between 10 and 500. Note that “batch size” below 10 are suggested to be avoided in order to speed up the training process. Hyperparameter of “learning rate” for all prediction tasks are produced in the interval  $[0.001, 0.1]$ . Table 5.8 gives the bounds of each hyperparameter.

### Other Settings

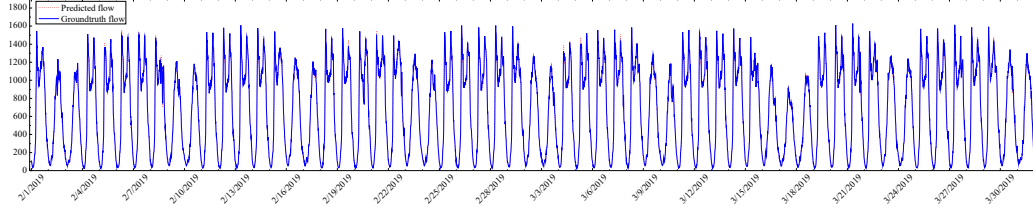
Other important parameters for PDE-LSTM are set as:  $(P, F_L, F_U, C_r, n, maxG) = (40, 0.5, 1, 0.7, 3, 10)$  for all of the datasets.

### 5.2.4 Prediction Performances Comparison

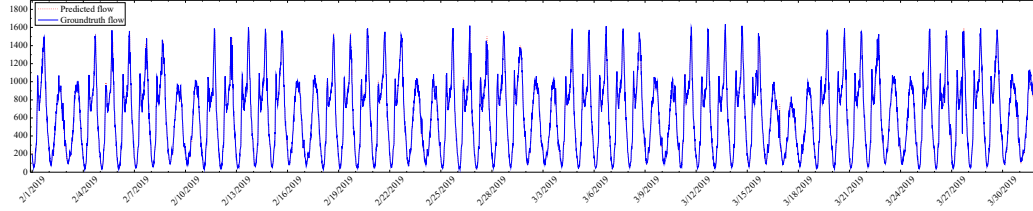
The prediction results on the 3 test sets by our proposed PDE-LSTM model are shown in Table S1, Table S2 and Table S3 in Supplementary file 5. Its mean accuracy (1-MAPE) is exceeding 93% for the four tasks on all of the test datasets (two thirds of them exceeding 94%). These results suggest that the forecasting accuracy obtained by PDE-LSTM is high, stable and robust.

Comparisons between the groundtruth flow and the predicted flow on the 15-min prediction task are presented in Fig. 5.3. The predicted traffic flow and trends by PDE-LSTM closely match with the groundtruth for all of the test sets.

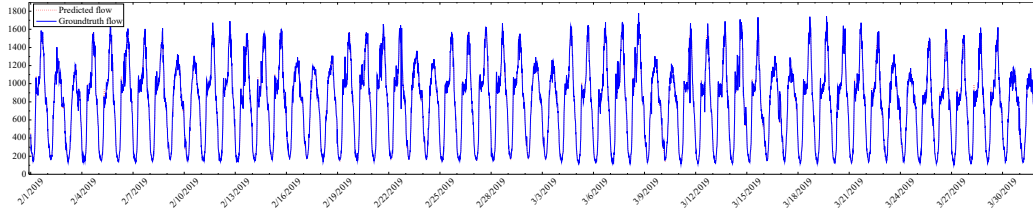
We also compare the performance of our proposed PDE-LSTM model with state-of-art baseline models in the field of traffic flow forecasting, including seasonal ARIMA (SARIMA), Gaussian process (GP), back-propagation neural network (BPNN) (Dougherty & Cobbett 1997), SVR



(a) M50-N (15-min)



(b) M1-N (15-min)



(c) I280-S (15-min)

Figure 5.3: Comparison between the predicted flow and the groundtruth flow on the 15-min prediction task. The predicted traffic flow and trends (denoted by red dash line) by our PDE-LSTM closely match with the groundtruth traffic flow (denoted by blue solid line) for all of the test sets.

(Yang et al. 2010) and deep-learning-based models like DNN (Qu et al. 2019), DBN (Huang et al. 2013, Huang et al. 2014), SAE (Lv et al. 2014), RNN, traditional LSTM (Tian & Pan 2015) and the LSTM under a random search strategy (denoted by LSTM (RSS)). To evaluate the effects of differential evolution algorithm on LSTM model, random search in a fixed set (Huang et al. 2014) is employed for LSTM (RSS) to make comparisons with PDE-LSTM. The fixed set is formed by the same hyperparameters of sequence length (i.e.,  $i = 1$ ), hidden unit (i.e.,  $i = 2$ ), maximum epoch (i.e.,  $i = 3$ ), batch size (i.e.,  $i = 4$ ) and learning rate (i.e.,  $i = 5$ ), ranging from 5, 5, 100, 50 and 0.001 to the corresponding upper bounds ( $b_{U_i}$  in PDE-LSTM as shown in table 5.8) with 5, 5, 100, 50 and 0.01 as gaps, respectively.

PDE-LSTM achieved the best performance in terms of MAE, RMSE or MAPE for all the prediction tasks on each dataset (see Table S1, Table S2 and Table S3 in Supplementary file 5). Methods such as SAE, DBN, RNN and LSTM exhibit a quite competitive accuracy to each other. SVR performs well in short-term predictions (e.g., 15-min prediction). However, its errors surge when conducting longer-term predictions (e.g., 30-min, 45-min and 60-min predictions). Overall, the deep learning models can achieve better performances than the other methods. The LSTM-based models can improve the accuracy in comparison with the other deep learning models. One reason is probably that LSTM is capable of learning long-term interdependencies (Liu, Shahroudy, Xu, Kot & Wang 2017), which makes it perform even better when modelling sequential time series data of traffic flow.

### 5.2.5 Residual and Correlation Analyses

Taking the 15-min prediction for example, Fig. 5.4 demonstrates the plot and distribution of residual errors on the test sets by PDE-LSTM. The residual errors are calculated using Eq. (5.22).

$$e_i = \hat{y}_i - y_i \quad (5.22)$$

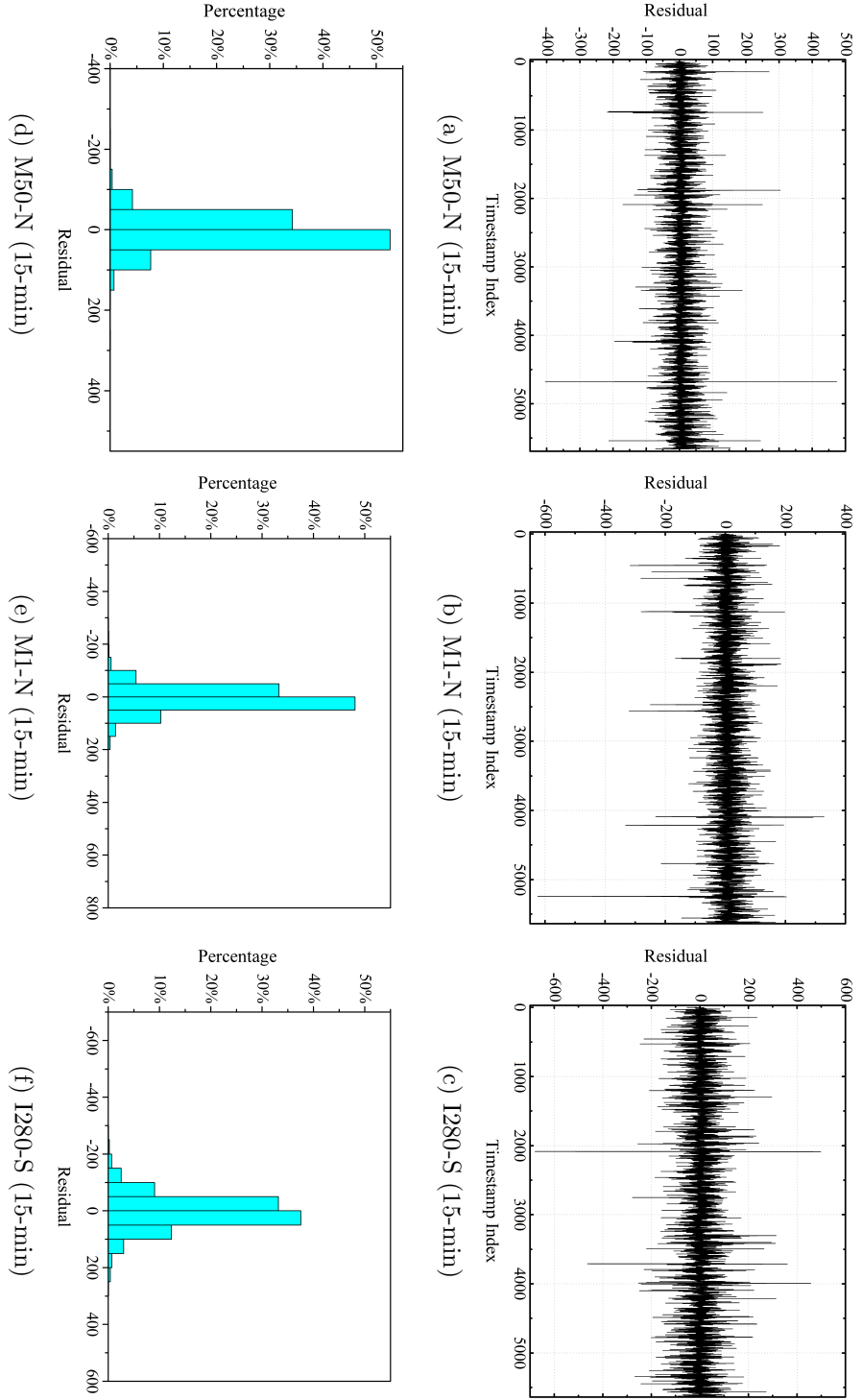


Figure 5.4: Residual error analysis for the 15-min prediction by PDE-LSTM model. (a)(b)(c) plot of residual error. (d)(e)(f) histogram of residual error

where  $\hat{y}_i$  is the  $i$ th groundtruth flow and  $y_i$  is the corresponding predicted flow. For the residuals in the M50-N dataset, 86.81% of them are distributed between  $[-50, 50)$ , and further 98.62% are covered by the interval of  $[-100, 100)$  (see Fig. 5.4 (d)). For the M1-N dataset, 81.37% of the residuals are located between  $[-50, 50)$ , and 97.09% of them are spanned between  $[-100, 100)$  (see Fig. 5.4 (e)). In a similar pattern, 70.71% and 92.09% of them are in  $[-50, 50)$  and  $[-100, 100)$ , respectively, for the residuals in the I280-S dataset (see Fig. 5.4 (f)).

Fig. S1 in Supplementary file 6 presents correlation plots between the groundtruth flow and the predicted flow by PDE-LSTM on the I280-S dataset (i.e., subfigures (a) (g) (m) and (s)). These correlations are also compared with those competitive and well-performed baselines, like LSTM (RSS), DBN, SVR, BPNN and SAE (see other subfigures in Fig. S1 of Supplementary file 6). The comparison shows that PDE-LSTM can cover more points inside the blue-lined area and less points outside for all of the four prediction tasks. This is another perspective to demonstrate that PDE-LSTM outperformed the baseline models.

### 5.2.6 Runtime Comparison

Parallel computing and early stopping strategy are implemented for PDE-LSTM to speed up the optimization process. The time costs by PDE-LSTM for all the prediction tasks are listed in Table 5.9. Overall, the required time for optimization increases with the increase of training data volume. The 15-min prediction takes the most amount of time (it was trained on the largest volume of training data). The computational costs by LSTM under the random search strategy (i.e., LSTM (RSS)) are also listed in Table 5.9. The optimization time for LSTM (RSS) may vary greatly even for the same prediction task (e.g., 19.78h vs 8.13h vs 6.97h for the 30-min prediction). One reason is probably that a large sequence length (or hidden units) or a small batch size are derived during the tuning phase. Overall, PDE-LSTM outperforms LSTM (RSS) in most of these prediction tasks in terms of time

efficiency. Even in the cases that PDE-LSTM takes more running time, the time costs by these two models have no significant differences (e.g., 9.93h vs 8.13h and 7.50h vs 6.97h in the 30-min prediction task).

Table 5.9: Computational time cost comparison

Dataset	Model	Optimization Time Cost (h)			
		15-min	30-min	45-min	60-min
M50-N	LSTM (RSS)	17.97	19.78	11.03	9.57
	PDE-LSTM	<b>11.02</b>	<b>5.77</b>	<b>5.35</b>	<b>2.42</b>
M1-N	LSTM (RSS)	20.68	<b>8.13</b>	7.47	10.25
	PDE-LSTM	<b>16.25</b>	9.93	<b>3.28</b>	<b>3.30</b>
I280-S	LSTM (RSS)	22.25	<b>6.97</b>	7.62	12.38
	PDE-LSTM	<b>12.83</b>	7.50	<b>4.63</b>	<b>3.17</b>

*Notes:* Our experiments were run on the server with 2.2GHz Intel Xeon Gold CPU 6238R with 26 cores enabled.

### 5.3 Summary

In this chapter, we developed a deep learning approach for urban traffic flow prediction. Unlike adopting traditional grid search or random search strategies, we use a differential evolution algorithm to determine the globally optimized hyperparameters for an LSTM network. Parallel computing is implemented in the approach to accelerate the optimization process. Our proposed method is named PDE-LSTM (parallel-differential-evolution-based LSTM). Experimental results evaluating on 3 real-world datasets show that PDE-LSTM has achieved at least 93% accuracies for all of the prediction tasks, and it has better performances than the state-of-the-art baseline methods: seasonal ARIMA, GP, BPNN, SVR, DNN, DBN, SAE, RNN and LSTM.



# Chapter 6

## Conclusion and Future Work

### 6.1 Conclusion

In this thesis, we have mainly addressed three research problems on traffic time series data analysis, namely taxi destination prediction, anomalous traffic patterns detection and urban traffic flow prediction. The proposed methods for solving these problems are detailed in Chapters 3-5. In the following content, the results and findings of each research problem are summarized.

We proposed a data-driven ensemble learning approach for the taxi destination prediction problem, which incorporates the advantages of SVR and DBN models for dealing with different segments of the trajectories. A novel data embedding technique named CFE was applied in deep learning model. We evaluated the individual and overall prediction performances and made comparisons with baselines of SVR, DBN, ANN, kNN and naive Bayes. From the experimental results on two real-world taxi GPS trajectory datasets collected from two independent urban cities, we demonstrated that our ensemble learning approach performs better than other models in terms of the overall performance. In general, it can get more accurate predictions, when the taxi is getting closer to the drop-off location. Experiments also showed the effectiveness of our proposed CFE technique in deep learning.

We developed novel methods for online and offline detections of anomalous traffic patterns from bus trajectory datasets. Our methods have explored deep learning ideas to extract novel features and the methods can make good visualization of the features as well. Based on the spatial and temporal characteristics of the anomalies, we have termed *class A anomaly* and *class B anomaly* to better address the discrepancy issues between these diversified anomalous patterns. The key idea behind our algorithms is to use the Boxplot rule or the nearest neighborhood for different detection tasks of anomalous patterns. Our methods are able to conduct insights analysis on the locations of anomalies as well as on the traffic influences to the road caused by the corresponding anomalies, after the visualized color trajectories are fused with GIS map to generate a color trajectory map. We also developed an online detection method extending from the offline method for a real-time anomalous traffic patterns detection. Comprehensive experiments on three real-world bus route datasets confirmed the effectiveness and superiority of our deep feature extraction method and anomaly detection approaches while comparing with the baseline methods PCA, RP, SSAE, one-class SVM, binary SVM, LSTM, HDBSCAN and kNN.

The thesis presented an improved deep learning approach for urban traffic flow prediction. The key idea is to use a differential evolution algorithm to determine globally optimized hyperparameters for an LSTM network, and the parameter search is implemented by parallel computing and early stopping to accelerate the optimization process. This search strategy for hyperparameter optimization is advanced to grid search or random search commonly adopted by other deep learning models. Our method is named PDE-LSTM (parallel-differential-evolution-based LSTM). The proposed PDE-LSTM model was evaluated on three real-world flow datasets to confirm its superior prediction performance in comparison with the baseline methods of seasonal ARIMA, GP, BPNN, SVR, DNN, DBN, SAE, RNN and LSTM. PDE-LSTM has achieved at least 93% accuracy for all of the prediction tasks. This result is expected, as an LSTM-based model with optimized hyperparameters is more

capable of learning sequential features than other deep learning models.

## **6.2 Future Work**

In addition to the encouraging results and findings, there are still some problems as well as challenges need to be addressed in the future work.

Firstly, for the problem of taxi destination prediction, we have constructed a classifier based on a simple lazy learning algorithm kNN to estimate the current segment of the whole trajectory. Although the kNN classifier can achieve a high overall accuracy on large-scale data, it delivered unsatisfactory performances when the percentage of the whole trajectory is around 20%, 30% or 40%. It would be interesting to investigate other promising methods, such as the state-of-the-art deep learning methods, to improve the generalization ability of classifier for providing a more accurate result. Furthermore, prediction of the arrival time of taxi would be also of great significance for this problem.

Secondly, the proposed online anomaly detection method leads a relatively high false alarm rate when tested on some datasets. One reason is probably that we chose the most similar color trajectory from the training set as the reference to construct a new color trajectory. Under the circumstances, it might be not robust to get a color trajectory that follows a similar pattern with the original trajectory. In this case, we will focus on developing a more effective and reliable method to measure the similarity between two color trajectories. Besides, we will explore the possibility of transferring our proposed methods to other trajectory data sources, such as the city-wide taxis or trains trajectory data.

Thirdly, the proposed PDE-LSTM model is designed for processing continuous variables, however, some discrete variables in LSTM network such as the loss function, the normalization function and time features embedding cannot be optimized directly via the proposed model. As future work, we will investigate how to make our proposed algorithm adapt to

discrete hyperparameters optimization. On the other hand, exploring other kinds of optimization methods in this problem and making comparisons with differential evolution will also be the next stage of this research.

# Appendix A

## Appendix: List of Supplementary Files

The supplementary file list and the corresponding download links

name	chapter	description	link
Supplementary file 1	3	supplementary tables for chapter 3	link
Supplementary file 2	4	supplementary tables for chapter 4	link
Supplementary file 3	4	supplementary figures for chapter 4	link
Supplementary file 4	4	data and code for chapter 4	link
Supplementary file 5	5	supplementary tables for chapter 5	link
Supplementary file 6	5	supplementary figures for chapter 5	link
Supplementary file 7	5	data and code for chapter 5	link

*Notes:* If there are any issues with the links, please visit [https://github.com/Xiaocai-Zhang/Thesis\\_backup](https://github.com/Xiaocai-Zhang/Thesis_backup) to download the above supplementary tables and figures.

# Appendix B

## Appendix: List of Abbreviations

The following list is neither exhaustive nor exclusive, but may be helpful.

ITS	Intelligent transportation systems
GPS	Global positioning system
LBSs	Location-based services
SVR	Support vector regression
RGB	Red-green-blue
GIS	Geographic information systems
HMM	Hidden Markov chain model
PCA	Principal component analysis
SVM	Support vector machine
kNN	$k$ -nearest neighbour
RNN	Recurrent neural network

LSTM	Long short-term memory
ARIMA	Auto-regressive integrated moving average
VARMA	Vector auto-regressive moving average
ES	Exponential smoothing
NN	Neural network
FNN	Fuzzy neural network
SAE	Stacked autoencoder
DBN	Deep belief network
DNN	Deep neural network
CNN	Convolutional neural network
GMM	Gaussian mixture model
BCM	Bayesian combination method
ELM	Ensemble learning model
CFE	Circular fuzzy embedding
MAE	Mean absolute error
RMSE	Root mean square error
AMAE	Average mean absolute error
ARMSE	Average root mean square error
RBF	Radial basis function
EV	Electric vehicle
ANN	Artificial neural network

NB	Naive Bayes
One-Hot-E	One-hot embedding
API	Application programming interface
DSAE	Deep sparse autoencoder
CT	Color trajectory
CTM	Color trajectory map
KL	Kullback-Leibler
Acc	Accuracy
DR	Detection rate
FAR	False alarm rate
ROC	Receiver operating characteristic
AUC	Area under the ROC curve
TP	True positive
TN	True negative
FP	False positive
FN	False negative
AMSD	Averaged moving standard deviation
SUV	Sport utility vehicle
RP	Random projection
SSAE	Single sparse autoencoder
OneSVM	One-class support vector machine



BiSVM	Binary SVM
DE	Differential evolution
SGD	Stochastic gradient decent
MAPE	Mean absolute percentage error
TII	Transport infrastructure Ireland
PeMS	Performance measurement system
GP	Gaussian process
BPNN	Back-propagation neural network
RSS	Random search strategy



# Bibliography

- Ahmed, M. S. & Cook, A. R. (1979), ‘Analysis of freeway traffic time-series data by using box-jenkins techniques’, *Transportation Research Record* (722), 1–9.
- Alvarez-Garcia, J. A., Ortega, J. A., Gonzalez-Abril, L. & Velasco, F. (Dec. 2010), ‘Trip destination prediction based on past gps log using a hidden markov model’, *Expert Systems with Applications* **37**(12), 8166–8171.
- An, J., Fu, L., Hu, M., Chen, W. & Zhan, J. (2019), ‘A novel fuzzy-based convolutional neural network method to traffic flow prediction with uncertain traffic accident information’, *IEEE Access* **7**, 20708–20722.
- Ashbrook, D. & Starner, T. (Sept. 2003), ‘Using gps to learn significant locations and predict movement across multiple users’, *Personal and Ubiquitous Computing* **7**(5), 275–286.
- Asif, M. T., Dauwels, J., Goh, C. Y., Oran, A., Fathi, E., Xu, M., Dhanya, M. M., Mitrovic, N. & Jaillet, P. (2013), ‘Spatiotemporal patterns in large-scale traffic speed prediction’, *IEEE Transactions on Intelligent Transportation Systems* **15**(2), 794–804.
- Barbará, D., Domeniconi, C. & Rogers, J. P. (2006), Detecting outliers using transduction and statistical testing, *in* ‘Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining’, ACM, pp. 55–64.

- Barria, J. A. & Thajchayapong, S. (2011), ‘Detection and classification of traffic anomalies using microscopic traffic variables’, *IEEE Transactions on Intelligent Transportation Systems* **12**(3), 695–704.
- Beijing Public Transport (2020), [http://www.bjbus.com/home/fun\\_static\\_page.php?uSec=00000156&uSub=00000157](http://www.bjbus.com/home/fun_static_page.php?uSec=00000156&uSub=00000157). Online; accessed 10 May 2020.
- Belenguer, F. M., Salcedo, A. M., Ibanez, A. G. & Sanchez, V. M. (2019), ‘Advantages offered by the double magnetic loops versus the conventional single ones’, *PloS one* **14**(2).
- Bergstra, J. & Bengio, Y. (2012), ‘Random search for hyper-parameter optimization’, *Journal of Machine Learning Research* **13**(Feb), 281–305.
- Bidasca, L. & Townsend, E. (2016), *Making taxis safer: managing road risks for taxi drivers, their passengers and other road users*, European Transport Safety Council, Brussels, Belgium, pp. 5–6.
- Callegari, C., Gazzarrini, L., Giordano, S., Pagano, M. & Pepe, T. (2011), A novel pca-based network anomaly detection, in ‘Proceedings of 2011 IEEE International Conference on Communications (ICC)’, IEEE, pp. 1–5.
- Campello, R. J., Moulavi, D. & Sander, J. (2013), Density-based clustering based on hierarchical density estimates, in ‘Proceedings of the 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining’, Springer, pp. 160–172.
- Castro-Neto, M., Jeong, Y.-S., Jeong, M.-K. & Han, L. D. (2009), ‘Online-svr for short-term traffic flow prediction under typical and atypical traffic conditions’, *Expert Systems with Applications* **36**(3), 6164–6173.
- Chandola, V., Banerjee, A. & Kumar, V. (2009), ‘Anomaly detection: A survey’, *ACM Computing Surveys (CSUR)* **41**(3), 15:1–15:58.

- Chauhan, S. & Vig, L. (2015), Anomaly detection in ecg time signals via deep long short-term memory networks, *in* ‘Proceedings of 2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)’, IEEE, pp. 1–7.
- Chawla, S., Zheng, Y. & Hu, J. (2012), Inferring the root cause in road traffic anomalies, *in* ‘Proceedings of 2012 IEEE International Conference on Data Mining’, IEEE, pp. 141–150.
- Chen, C., Zhang, D., Castro, P. S., Li, N., Sun, L. & Li, S. (2011), Real-time detection of anomalous taxi trajectories from gps traces, *in* ‘Proceedings of 8th International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services’, Springer, pp. 63–74.
- Chen, M., Liu, Y. & Yu, X. (2014), Nlpmm: A next location predictor with markov modeling, *in* ‘Proceedings of the 18th Pacific-Asia Conference on Knowledge Discovery and Data Mining’, Springer, pp. 186–197.
- Costa, V., Fontes, T., Costa, P. M. & Dias, T. G. (2015), Prediction of journey destination in urban public transport, *in* ‘Proceedings of the 17th Portuguese Conference on Artificial Intelligence’, pp. 169–180.
- Dai, J., Yang, B., Guo, C. & Ding, Z. (2015), Personalized route recommendation using big trajectory data, *in* ‘Proceedings of the 2015 IEEE 31st International Conference on Data Engineering’, IEEE, pp. 543–554.
- Dai, X. & Zhu, Y. (2018), ‘Towards theoretical understanding of large batch training in stochastic gradient descent’, *arXiv preprint arXiv:1812.00542*.
- DataCastle (2016), <https://www.dcjingsai.com>. Online; accessed 30 Sep 2017.
- David, O. E. & Netanyahu, N. S. (2015), Deepsign: deep learning for automatic malware signature generation and classification, *in*

- ‘Proceedings of 2015 International Joint Conference on Neural Networks (IJCNN)’, pp. 1–8.
- Davis, G. A. & Nihan, N. L. (1991), ‘Nonparametric regression and short-term freeway traffic forecasting’, *Journal of Transportation Engineering* **117**(2), 178–188.
- De Brébisson, A., Simon, É., Auvolet, A., Vincent, P. & Bengio, Y. (2015), ‘Artificial neural networks applied to taxi destination prediction’, *arXiv preprint arXiv:1508.00021* .
- Dettling, M. (2013), *Applied time series analysis*, Swiss Federal Institute of Technology Zurich, p. 11.
- Ding, Q. Y., Wang, X. F., Zhang, X. Y. & Sun, Z. Q. (2011), ‘Forecasting traffic volume with space-time arima model’, *Advanced Materials Research* **156**, 979–983.
- Ding, Y., Liu, S., Pu, J. & Ni, L. M. (2013), Hunts: a trajectory recommendation system for effective and efficient hunting of taxi passengers, *in* ‘Proceedings of 2013 IEEE International Conference on Mobile Data Management’, pp. 107–116.
- Dougherty, M. S. & Cobbett, M. R. (1997), ‘Short-term inter-urban traffic forecasts using neural networks’, *International Journal of Forecasting* **13**(1), 21–31.
- Duan, P. (2019), Modeling, Analysis and Application of Big Traffic Data for Intelligent Transportation Systems, PhD dissertation, University of Technology Sydney.
- Fan, L. & Xiong, L. (2013), Differentially private anomaly detection with a case study on epidemic outbreak detection, *in* ‘Proceedings of the IEEE 13th International Conference on Data Mining Workshops’, IEEE, pp. 833–840.

- Feng, T. & Timmermans, H. J. (2013), ‘Transportation mode recognition using gps and accelerometer data’, *Transportation Research Part C: Emerging Technologies* **37**, 118–130.
- Fontugne, R., Abry, P., Fukuda, K., Borgnat, P., Mazel, J., Wendt, H. & Veitch, D. (2015), Random projection and multiscale wavelet leader based anomaly detection and address identification in internet traffic, *in* ‘Proceedings of 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)’, IEEE, pp. 5530–5534.
- Gal, Y. (2016), Uncertainty in deep learning, PhD dissertation, University of Cambridge.
- Gambs, S., Killijian, M.-O. & del Prado Cortez, M. N. (2012), Next place prediction using mobility markov chains, *in* ‘Proceedings of the First Workshop on Measurement, Privacy, and Mobility’, p. 3.
- Gao, N., Gao, L., Gao, Q. & Wang, H. (2014), An intrusion detection model based on deep belief networks, *in* ‘Proceedings of 2014 Second International Conference on Advanced Cloud and Big Data’, pp. 247–252.
- Guo, J., Huang, W. & Williams, B. M. (2014), ‘Adaptive kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification’, *Transportation Research Part C: Emerging Technologies* **43**(1), 50–64.
- Hamza, N. M., Essam, D. L. & Sarker, R. A. (2015), ‘Constraint consensus mutation-based differential evolution for constrained optimization’, *IEEE Transactions on Evolutionary Computation* **20**(3), 447–459.
- Hautamaki, V., Karkkainen, I. & Franti, P. (2004), Outlier detection using k-nearest neighbour graph, *in* ‘Proceedings of the 17th International Conference on Pattern Recognition’, IEEE, pp. 430–433.

- Hinton, G. E. (Mar. 2002), ‘Training products of experts by minimizing contrastive divergence’, *Neural Computation* **14**(8), 1771–1800.
- Hinton, G. E., Osindero, S. & Teh, Y.-W. (May 2006), ‘A fast learning algorithm for deep belief nets’, *Neural Computation* **18**(7), 1527–1554.
- Hodge, V. J., Krishnan, R., Austin, J., Polak, J. & Jackson, T. (2014), ‘Short-term prediction of traffic flow using a binary neural network’, *Neural Computing and Applications* **25**(7-8), 1639–1655.
- Hong, W.-C., Dong, Y., Zheng, F. & Lai, C.-Y. (2011), ‘Forecasting urban traffic flow by svr with continuous aco’, *Applied Mathematical Modelling* **35**(3), 1282–1291.
- Huang, W., Hong, H., Li, M., Hu, W., Song, G. & Xie, K. (2013), Deep architecture for traffic flow prediction, in ‘Proceedings of the 9th International Conference on Advanced Data Mining and Applications’, Springer, pp. 165–176.
- Huang, W., Song, G., Hong, H. & Xie, K. (2014), ‘Deep architecture for traffic flow prediction: deep belief networks with multitask learning’, *IEEE Transactions on Intelligent Transportation Systems* **15**(5), 2191–2201.
- Ji, N., Zhang, J., Zhang, C. & Wang, L. (Aug. 2014), ‘Discriminative restricted boltzmann machine for invariant pattern recognition with linear transformations’, *Pattern Recognition Letters* **45**, 172–180.
- Jia, Y., Wu, J. & Xu, M. (2017), ‘Traffic flow prediction with rainfall impact using a deep learning method’, *Journal of Advanced Transportation* **2017**, 1–11.
- Jieni, X. & Zhongke, S. (2008), ‘Short-time traffic flow prediction based on chaos time series theory’, *Journal of Transportation Systems Engineering and Information Technology* **8**(5), 68–72.



- Juvonen, A. & Hamalainen, T. (2014), An efficient network log anomaly detection system using random projection dimensionality reduction, *in* ‘Proceedings of the 6th International Conference on New Technologies, Mobility and Security (NTMS)’, IEEE, pp. 1–5.
- Kaggle (2015), ‘ECML/PKDD 15: Taxi Trajectory Prediction’, <https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i>. Online; accessed 30 May 2017.
- Kang, D., Lv, Y. & Chen, Y.-y. (2017), Short-term traffic flow prediction with lstm recurrent neural network, *in* ‘Proceedings of the 2017 IEEE International Conference on Intelligent Transportation Systems’, IEEE, pp. 1–6.
- Kim, T.-Y. & Cho, S.-B. (2018), ‘Web traffic anomaly detection using c-lstm neural networks’, *Expert Systems with Applications* **106**, 66–76.
- Kong, X., Song, X., Xia, F., Guo, H., Wang, J. & Tolba, A. (2018), ‘Lotad: Long-term traffic anomaly detection based on crowdsourced bus trajectory data’, *World Wide Web* **21**(3), 825–847.
- Kuang, W., An, S. & Jiang, H. (2015), ‘Detecting traffic anomalies in urban areas using taxi gps data’, *Mathematical Problems in Engineering* **2015**, 1–14.
- Kumar, S. V. & Vanajakshi, L. (2015), ‘Short-term traffic flow prediction using seasonal arima model with limited input data’, *European Transport Research Review* **7**(3), 1–9.
- Lakhina, A., Crovella, M. & Diot, C. (2004), ‘Diagnosing network-wide traffic anomalies’, *ACM SIGCOMM Computer Communication Review* **34**(4), 219–230.
- Le, T. V., Liu, S. & Lau, H. C. (2016), Reinforcement learning framework for modeling spatial sequential decisions under uncertainty, *in* ‘Proceedings

of the 15th International Conference on Autonomous Agents and Multiagent Systems’, pp. 1449–1450.

LeCun, Y., Bengio, Y. & Hinton, G. (2015), ‘Deep learning’, *Nature* **521**(7553), 436–444.

Leduc, G. et al. (2008), ‘Road traffic data: Collection methods and applications’, *Working Papers on Energy, Transport and Climate Change* **1**(55), 1–55.

Lee, S. & Fambro, D. B. (1999), ‘Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting’, *Transportation Research Record* **1678**(1), 179–188.

Leung, K. & Leckie, C. (2005), Unsupervised anomaly detection in network intrusion detection using clusters, in ‘Proceedings of the Twenty-eighth Australasian Conference on Computer Science’, Australian Computer Society, Inc., pp. 333–342.

Li, H. (2016), ‘Research on prediction of traffic flow based on dynamic fuzzy neural networks’, *Neural Computing and Applications* **27**(7), 1969–1980.

Li, K.-L., Huang, H.-K., Tian, S.-F. & Xu, W. (2003), Improving one-class svm for anomaly detection, in ‘Proceedings of the 2003 International Conference on Machine Learning and Cybernetics’, IEEE, pp. 3077–3081.

Li, L., Su, X., Zhang, Y., Lin, Y. & Li, Z. (2015), ‘Trend modeling for traffic time series analysis: An integrated study’, *IEEE Transactions on Intelligent Transportation Systems* **16**(6), 3430–3439.

Li, M., Ahmed, A. & Smola, A. J. (2015), Inferring movement trajectories from gps snippets, in ‘Proceedings of the Eighth ACM International Conference on Web Search and Data Mining’, pp. 325–334.

- Li, Y., Guo, T., Xia, R. & Xie, W. (2018), ‘Road traffic anomaly detection based on fuzzy theory’, *IEEE Access* **6**, 40281–40288.
- Li, Y., Liu, W. & Huang, Q. (2016), ‘Traffic anomaly detection based on image descriptor in videos’, *Multimedia Tools and Applications* **75**(5), 2487–2505.
- Lin, H.-T., Lin, C.-J. & Weng, R. C. (2007), ‘A note on platt’s probabilistic outputs for support vector machines’, *Machine Learning* **68**(3), 267–276.
- Lin, S.-L., Huang, H.-Q., Zhu, D.-Q. & Wang, T.-Z. (2009), The application of space-time arima model on traffic flow forecasting, in ‘Proceedings of the 2009 International Conference on Machine Learning and Cybernetics’, IEEE, pp. 3408–3412.
- Ling, C. X. (1995), ‘Overfitting and generalization in learning discrete patterns’, *Neurocomputing* **8**(3), 341–347.
- Liu, H., Taniguchi, T., Tanaka, Y., Takenaka, K. & Bando, T. (2017), ‘Visualization of driving behavior based on hidden feature extraction by using deep learning’, *IEEE Transactions on Intelligent Transportation Systems* **18**(9), 2477–2489.
- Liu, J., Shahroudy, A., Xu, D., Kot, A. C. & Wang, G. (2017), ‘Skeleton-based action recognition using spatio-temporal lstm network with trust gates’, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **40**(12), 3007–3021.
- Liu, S., Ni, L. M. & Krishnan, R. (Jan. 2014), ‘Fraud detection from taxis’ driving behaviors’, *IEEE Transactions on Vehicular Technology* **63**(1), 464–472.
- Liu, S. & Qu, Q. (2016), ‘Dynamic collective routing using crowdsourcing data’, *Transportation Research Part B: Methodological* **93**, 450–469.

- Liu, S. & Wang, S. (2017), ‘Trajectory community discovery and recommendation by multi-source diffusion modeling’, *IEEE Transactions on Knowledge and Data Engineering* **29**(4), 898–911.
- Liu, W., Zheng, Y., Chawla, S., Yuan, J. & Xing, X. (2011), Discovering spatio-temporal causal interactions in traffic data streams, *in* ‘Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining’, ACM, pp. 1010–1018.
- Liu, X., Gong, L., Gong, Y. & Liu, Y. (Feb. 2015), ‘Revealing travel patterns and city structure with taxi trip data’, *Journal of Transport Geography* **43**, 78–90.
- Liu, Y., Zhang, L. & Guan, Y. (2010), Sketch-based streaming pca algorithm for network-wide traffic anomaly detection, *in* ‘Proceedings of the 30th International Conference on Distributed Computing Systems’, IEEE, pp. 807–816.
- Luo, X., Li, D., Yang, Y. & Zhang, S. (2019), ‘Spatiotemporal traffic flow prediction with knn and lstm’, *Journal of Advanced Transportation* **2019**, 1–11.
- Lv, P., Yu, Y., Fan, Y., Tang, X. & Tong, X. (2020), ‘Layer-constrained variational autoencoding kernel density estimation model for anomaly detection’, *Knowledge-Based Systems* **196**, 105753.
- Lv, Y., Duan, Y., Kang, W., Li, Z. & Wang, F.-Y. (2014), ‘Traffic flow prediction with big data: a deep learning approach’, *IEEE Transactions on Intelligent Transportation Systems* **16**(2), 865–873.
- Malhotra, P., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P. & Shroff, G. (2016), ‘Lstm-based encoder-decoder for multi-sensor anomaly detection’, *arXiv preprint arXiv:1607.00148*.

- Manasseh, C. & Sengupta, R. (2013), Predicting driver destination using machine learning techniques, *in* ‘Proceedings of the 2013 International IEEE Conference on Intelligent Transportation Systems’, pp. 142–147.
- Mao, J., Sun, P., Jin, C. & Zhou, A. (2018), Outlier detection over distributed trajectory streams, *in* ‘Proceedings of the 2018 SIAM International Conference on Data Mining’, SIAM, pp. 64–72.
- Mihaita, A.-S., Li, H., He, Z. & Rizoio, M.-A. (2019), Motorway traffic flow prediction using advanced deep learning, *in* ‘Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference’, IEEE, pp. 1683–1690.
- Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013), ‘Efficient estimation of word representations in vector space’, *arXiv preprint arXiv:1301.3781*.
- Min, W. & Wynter, L. (2011), ‘Real-time road traffic prediction with spatio-temporal correlations’, *Transportation Research Part C: Emerging Technologies* **19**(4), 606–616.
- Mohamed, A.-r., Dahl, G. E. & Hinton, G. (Jan. 2012), ‘Acoustic modeling using deep belief networks’, *IEEE Transactions on Audio, Speech, and Language Processing* **20**(1), 14–22.
- Münz, G., Li, S. & Carle, G. (2007), Traffic anomaly detection using k-means clustering, *in* ‘GI/ITG Workshop MMBnet’, pp. 13–14.
- Nguyen, H., Liu, W., Rivera, P. & Chen, F. (2016), Trafficwatch: Real-time traffic incident detection and monitoring using social media, *in* ‘Proceedings of the 20th Pacific-Asia Conference on Knowledge Discovery and Data Mining’, Springer, pp. 540–551.
- Pan, T., Sumalee, A., Zhong, R.-X. & Indra-Payoong, N. (2013), ‘Short-term traffic state prediction based on temporal–spatial correlation’, *IEEE Transactions on Intelligent Transportation Systems* **14**(3), 1242–1254.

- Pang, L., Chawla, S., Liu, W. & Zheng, Y. (2013), ‘On detection of emerging anomalous traffic patterns using gps data’, *Data & Knowledge Engineering* **87**, 357–373.
- Pang, L. X., Chawla, S., Liu, W. & Zheng, Y. (2011), On mining anomalous patterns in road traffic streams, in ‘Proceedings of 7th International Conference on Advanced Data Mining and Applications’, Springer, pp. 237–251.
- Pawling, A., Chawla, N. V. & Madey, G. (2007), ‘Anomaly detection in a mobile communication network’, *Computational and Mathematical Organization Theory* **13**(4), 407–422.
- PeMS (2019), <http://pems.dot.ca.gov/>. Online; accessed 15 Oct 2019.
- Peng, C., Jin, X., Wong, K.-C., Shi, M. & Liò, P. (Apr. 2012), ‘Collective human mobility pattern from taxi trips in urban area’, *PloS one* **7**(4), e34487.
- Peng, T., Liu, Q. & Wang, G. (Mar. 2017), ‘Enhanced location privacy preserving scheme in location-based services’, *IEEE Systems Journal* **11**(1), 219–230.
- Polson, N. G. & Sokolov, V. O. (2018), ‘Deep learning’, *arXiv preprint arXiv:1807.07987*.
- Provost, F. (2000), Machine learning from imbalanced data sets 101, in ‘Proceedings of the AAAI’2000 Workshop Imbalanced Data Sets’, AAAI Press, pp. 1–3.
- Qu, L., Li, W., Li, W., Ma, D. & Wang, Y. (2019), ‘Daily long-term traffic flow forecasting based on a deep neural network’, *Expert Systems with Applications* **121**, 304–312.

- Riveiro, M., Lebram, M. & Elmer, M. (2017), ‘Anomaly detection for road traffic: A visual analytics framework’, *IEEE Transactions on Intelligent Transportation Systems* **18**(8), 2260–2270.
- Rogers, J. P., Barbará, D. & Domeniconi, C. (2009), Detecting spatio-temporal outliers with kernels and statistical testing, *in* ‘Proceedings of the 17th International Conference on Geoinformatics’, IEEE, pp. 1–6.
- Rossi, A., Barlacchi, G., Bianchini, M. & Lepri, B. (2020), ‘Modelling taxi drivers’ behaviour for the next destination prediction’, *IEEE Transactions on Intelligent Transportation Systems* **21**(7), 2980–2989.
- Sakurada, M. & Yairi, T. (2014), Anomaly detection using autoencoders with nonlinear dimensionality reduction, *in* ‘Proceedings of the 2nd Workshop on Machine Learning for Sensory Data Analysis’, ACM, pp. 1–8.
- Schiller, J. & Voisard, A. (2004), *Location-based services*, Morgan Kaufmann, San Francisco, CA, USA, pp. 1–4.
- Simmons, R., Browning, B., Zhang, Y. & Sadekar, V. (2006), Learning to predict driver route and destination intent, *in* ‘Proceedings of the 2006 IEEE Intelligent Transportation Systems Conference’, pp. 127–132.
- Smola, A. J. & Schölkopf, B. (Aug. 2004), ‘A tutorial on support vector regression’, *Statistics and Computing* **14**(3), 199–222.
- Song, L., Wang, R., Xiao, D., Han, X., Cai, Y. & Shi, C. (2018), Anomalous trajectory detection using recurrent neural network, *in* ‘Proceedings of 14th International Conference on Advanced Data Mining and Applications’, Springer, pp. 263–277.
- Storn, R. & Price, K. (1997), ‘Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces’, *Journal of Global Optimization* **11**(4), 341–359.

- Sun, S. & Xu, X. (2010), ‘Variational inference for infinite mixtures of gaussian processes with applications to traffic flow prediction’, *IEEE Transactions on Intelligent Transportation Systems* **12**(2), 466–475.
- Sun, S., Zhang, C. & Yu, G. (2006), ‘A bayesian network approach to traffic flow forecasting’, *IEEE Transactions on Intelligent Transportation Systems* **7**(1), 124–132.
- Tan, M.-C., Wong, S. C., Xu, J.-M., Guan, Z.-R. & Zhang, P. (2009), ‘An aggregation approach to short-term traffic flow prediction’, *IEEE Transactions on Intelligent Transportation Systems* **10**(1), 60–69.
- Tang, J., Liu, F., Zou, Y., Zhang, W. & Wang, Y. (2017), ‘An improved fuzzy neural network for traffic speed prediction considering periodic characteristic’, *IEEE Transactions on Intelligent Transportation Systems* **18**(9), 2340–2350.
- Teevan, J., Karlson, A., Amini, S., Brush, A. & Krumm, J. (2011), Understanding the importance of location, time, and people in mobile local search behavior, *in* ‘Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services’, pp. 77–80.
- Teh, Y. W. & Hinton, G. E. (2001), Rate-coded restricted boltzmann machines for face recognition, *in* ‘Proceedings of the Fifteenth Conference on Neural Information Processing Systems’, pp. 908–914.
- Tian, Y. & Pan, L. (2015), Predicting short-term traffic flow by long short-term memory recurrent neural network, *in* ‘Proceedings of the 2015 IEEE international Conference on Smart City/SocialCom/SustainCom (SmartCity)’, IEEE, pp. 153–158.
- Tian, Y., Zhang, K., Li, J., Lin, X. & Yang, B. (2018), ‘Lstm-based traffic flow prediction with missing data’, *Neurocomputing* **318**, 297–305.



- Tian, Z., Jung, T., Wang, Y., Zhang, F., Tu, L., Xu, C., Tian, C. & Li, X.-Y. (2016), ‘Real-time charging station recommendation system for electric-vehicle taxis’, *IEEE Transactions on Intelligent Transportation Systems* **17**(11), 3098–3109.
- Tieleman, T. & Hinton, G. (2012), ‘Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude’, *COURSERA: Neural Networks for Machine Learning* **4**(2), 26–31.
- Transport Infrastructure Ireland (2019), <https://www.nratrafficdata.ie>. Online; accessed 10 Apr 2019.
- Tsai, C.-F. & Lin, C.-Y. (2010), ‘A triangle area based nearest neighbors approach to intrusion detection’, *Pattern Recognition* **43**(1), 222–229.
- U.S. Department of Transportation (2014), ‘A Summary of Vehicle Detection and Surveillance Technologies use in Intelligent Transportation Systems’, <https://www.fhwa.dot.gov/policyinformation/pubs/vdstits2007/03.cfm>. Online; accessed 9 May 2020.
- Van Der Voort, M., Dougherty, M. & Watson, S. (1996), ‘Combining kohonen maps with arima time series models to forecast traffic flow’, *Transportation Research Part C: Emerging Technologies* **4**(5), 307–318.
- Wang, H., Wen, H., Yi, F., Zhu, H. & Sun, L. (2017), ‘Road traffic anomaly detection via collaborative path inference from gps snippets’, *Sensors* **17**(3), 1–21.
- Wang, J., Deng, W. & Guo, Y. (2014), ‘New bayesian combination method for short-term traffic flow forecasting’, *Transportation Research Part C: Emerging Technologies* **43**(1), 79–94.
- Wang, P., Xu, B., Xu, J., Tian, G., Liu, C.-L. & Hao, H. (Jan. 2016), ‘Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification’, *Neurocomputing* **174**, 806–814.

- Wang, Y., Wong, J. & Miner, A. (2004), Anomaly intrusion detection using one class svm, *in* ‘Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop’, IEEE, pp. 358–364.
- Wang, Y., Zheng, Y. & Xue, Y. (2014), Travel time estimation of a path using sparse trajectories, *in* ‘Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining’, pp. 25–34.
- Wang, Z., Lu, M., Yuan, X., Zhang, J. & Van De Wetering, H. (2013), ‘Visual traffic jam analysis based on trajectory data’, *IEEE Transactions on Visualization and Computer Graphics* **19**(12), 2159–2168.
- Wu, C.-H., Ho, J.-M. & Lee, D.-T. (Dec. 2004), ‘Travel-time prediction with support vector regression’, *IEEE Transactions on Intelligent Transportation Systems* **5**(4), 276–281.
- Wu, G., Shen, X., Li, H., Chen, H., Lin, A. & Suganthan, P. N. (2018), ‘Ensemble of differential evolution variants’, *Information Sciences* **423**, 172–186.
- Wu, H., Sun, W. & Zheng, B. (2017), A fast trajectory outlier detection approach via driving behavior modeling, *in* ‘Proceedings of the 26th ACM International Conference on Information and Knowledge Management’, ACM, pp. 837–846.
- Wu, Y. & Tan, H. (2016), ‘Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework’, *arXiv preprint arXiv:1612.01022*.
- Xie, Y., Zhao, K., Sun, Y. & Chen, D. (2010), ‘Gaussian processes for short-term traffic volume forecasting’, *Transportation Research Record* **2165**(1), 69–78.
- Xu, M., Wang, D. & Li, J. (2016), Destpre: a data-driven approach to destination prediction for taxi rides, *in* ‘Proceedings of the 2016

- ACM International Joint Conference on Pervasive and Ubiquitous Computing’, pp. 729–739.
- Xu, X., Zhou, J., Liu, Y., Xu, Z. & Zhao, X. (2014), ‘Taxi-rs: Taxi-hunting recommendation system based on taxi gps data’, *IEEE Transactions on Intelligent Transportation Systems* **16**(4), 1716–1727.
- Xu, Y., Ouyang, X., Cheng, Y., Yu, S., Xiong, L., Ng, C.-C., Pranata, S., Shen, S. & Xing, J. (2018), Dual-mode vehicle motion pattern learning for high performance road traffic anomaly detection, *in* ‘Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops’, pp. 145–152.
- Xue, A. Y., Zhang, R., Zheng, Y., Xie, X., Huang, J. & Xu, Z. (2013), Destination prediction by sub-trajectory synthesis and privacy protection against such prediction, *in* ‘Proceedings of the 2013 IEEE International Conference on Data Engineering (ICDE)’, pp. 254–265.
- Yang, B., Sun, S., Li, J., Lin, X. & Tian, Y. (2019), ‘Traffic flow prediction using lstm with feature enhancement’, *Neurocomputing* **332**, 320–327.
- Yang, H.-F., Dillon, T. S. & Chen, Y.-P. P. (2016), ‘Optimized structure of the traffic flow forecasting model with a deep learning approach’, *IEEE Transactions on Neural Networks and Learning Systems* **28**(10), 2371–2381.
- Yang, J., Tan, G., Wang, F., Tian, Z. & Pan, D. (2010), ‘A parallel svr approach to real-time traffic flow forecasting’, *Journal of Dalian University of Technology* **50**, 1035–1041.
- Yin, H., Wong, S., Xu, J. & Wong, C. (2002), ‘Urban traffic flow prediction using a fuzzy-neural approach’, *Transportation Research Part C: Emerging Technologies* **10**(2), 85–98.
- Yu, Y., Cao, L., Rundensteiner, E. A. & Wang, Q. (2014), Detecting moving object outliers in massive-scale trajectory streams, *in* ‘Proceedings of the

20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', ACM, pp. 422–431.

Zadeh, L. A. (June 1965), 'Fuzzy sets', *Information and Control* **8**(3), 338–353.

Zhang, C., Song, D., Chen, Y., Feng, X., Lumezanu, C., Cheng, W., Ni, J., Zong, B., Chen, H. & Chawla, N. V. (2019), A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data, *in* 'Proceedings of the AAAI Conference on Artificial Intelligence', Vol. 33, pp. 1409–1416.

Zhang, D., Li, N., Zhou, Z.-H., Chen, C., Sun, L. & Li, S. (2011), ibat: detecting anomalous taxi trajectories from gps traces, *in* 'Proceedings of the 13th International Conference on Ubiquitous Computing', ACM, pp. 99–108.

Zhang, J., Wang, F.-Y., Wang, K., Lin, W.-H., Xu, X. & Chen, C. (2011), 'Data-driven intelligent transportation systems: A survey', *IEEE Transactions on Intelligent Transportation Systems* **12**(4), 1624–1639.

Zhang, X., Liu, Y., Zheng, Y., Zhao, Z., Li, J. & Liu, Y. (2018), Distinction between ships and icebergs in sar images using ensemble loss trained convolutional neural networks, *in* 'Proceedings of the 31st Australasian Joint Conference on Artificial Intelligence', Springer, pp. 216–223.

Zhang, X., Zhang, X., Verma, S., Liu, Y., Blumenstein, M. & Li, J. (2019), Detection of anomalous traffic patterns and insight analysis from bus trajectory data, *in* 'Proceedings of the 16th Pacific Rim International Conference on Artificial Intelligence', Springer, pp. 307–321.

Zhang, X., Zhao, Z., Zheng, Y. & Li, J. (2019), 'Prediction of taxi destinations using a novel data embedding method and ensemble

- learning', *IEEE Transactions on Intelligent Transportation Systems* **21**(1), 68–78.
- Zhao, J. & Sun, S. (2016), 'High-order gaussian process dynamical models for traffic flow prediction', *IEEE Transactions on Intelligent Transportation Systems* **17**(7), 2014–2019.
- Zhao, J., Yi, Z., Pan, S., Zhao, Y., Zhao, Z., Su, F. & Zhuang, B. (2019), Unsupervised traffic anomaly detection using trajectories, in 'Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops', pp. 133–140.
- Zhu, J. Z., Cao, J. X. & Zhu, Y. (2014), 'Traffic volume forecasting based on radial basis function neural network with the consideration of traffic flows at the adjacent intersections', *Transportation Research Part C: Emerging Technologies* **47**(2), 139–154.
- Zhu, L., Yu, F. R., Wang, Y., Ning, B. & Tang, T. (2018), 'Big data analytics in intelligent transportation systems: A survey', *IEEE Transactions on Intelligent Transportation Systems* **20**(1), 383–398.
- Zhu, T., Lin, Y. & Liu, Y. (2020), 'Improving interpolation-based oversampling for imbalanced data learning', *Knowledge-Based Systems* **187**, 104826.
- Ziebart, B. D., Maas, A. L., Dey, A. K. & Bagnell, J. A. (2008), Navigate like a cabbie: probabilistic reasoning from observed context-aware behavior, in 'Proceedings of the 10th International Conference on Ubiquitous Computing', pp. 322–331.

