Machine learning methods for transportation under uncertainty

Peled, Inon

2021

Peled, I. (2021). Machine learning methods for transportation under uncertainty. Doctoral thesis, Nanyang Technological University, Singapore. <https://hdl.handle.net/10356/153581>

<https://hdl.handle.net/10356/153581>

<https://doi.org/10.32657/10356/153581>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 18:34:10 SGT

Machine Learning Methods for Transportation under Uncertainty

Inon Peled

Interdisciplinary Graduate School

A thesis submitted to the Nanyang Technological University in partial fulfillment of the requirements for the degree of Doctor of Philosophy

2021

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution other than Nanyang Technological University and the Technical University of Denmark, as part of their joint PhD program.

4-Dec-2021

.....

Date

NTU NTU NTU NTU NTU NTU NTU
NTU NTU NTU NTU NTU NTU NTU
NTU NTU NTU NTU NTU NTU NTU
NTU NTU NTU NTU NTU NTU NTU

.....

Inon Peled

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

4-Dec-2021

.....
Date

.....
Justin Dauwels

.....
Francisco C. Pereira

Authorship Attribution Statement

This thesis contains material from 4 papers published in peer-reviewed journals and 1 paper accepted at a conference in which I am listed as an author, as follows.

Chapter 2 is published as Peled, I., Kamalakar, R., Azevedo, C. L., Pereira, F. C. (2020). QTIP: Quick simulation-based adaptation of traffic model per incident parameters. *Journal of Simulation*, 1-21. DOI: 10.1080/17477778.2020.1756702

The contributions of the co-authors to Chapter 2 are as follows:

- Raghuveer Kamalakar created the basis for this work in his MSc thesis.
- Profs. Carlos L. Azevedo and Francisco C. Pereira provided guidance for this work, reviewed it and confirmed the evaluation of its results.
- Prof. Carlos L. Azevedo also collected additional data on accidents in the concerned road segment and visualized this data.
- I extended the topic of co-author Raghuveer Kamalakar's MSc thesis into a conceptual framework for real-time model adaptation upon road incidents.
- I conducted and analyzed multiple experiments, including model fitting and transfer learning.
- I wrote the great majority of the work and revised it several times according to peer review comments.

Chapter 3 is published as Gammelli, D., Peled, I., Rodrigues, F., Pacino, D., Kurtaran, H. A., Pereira, F. C. (2020). Estimating latent demand of shared mobility through censored gaussian processes. *Transportation Research Part C: Emerging Technologies*, Vol. 120. DOI: 10.1016/j.trc.2020.102775

The contributions of the co-authors to Chapter 3 are as follows:

- I planned and implemented the stochastic censorship methodology in Section 4.5 (Taxi Demand and Supply).

- I carried out the experiments in Section 4.5 (Taxi Demand and Supply) and analyzed their results. In response to reviewers, I also conducted and analyzed experiments with matrix-based imputation.
- In addition to reviewing the entire manuscript, I wrote Section 2.2 (Censored Modeling), Section 4.5 (Taxi Demand and Supply) and parts of Sections 1 (Introduction) and 5 (Summary and Future Work).
- As the main author of the work, Daniele Gammelli wrote the remaining Sections and conducted the experiments in them.
- Profs. Filipe Rodrigues, Dario Pacino and Francisco C. Pereria guided the work, reviewed it and confirmed the evaluation of its results.

Research in Chapter 5 is published as Peled, I., Lee, K., Jiang, Y., Dauwels, J., Pereira, F. C. (2019). Preserving uncertainty in demand prediction for autonomous mobility services. *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 3043-3048. DOI: 10.1109/ITSC.2019.8916878

The contributions of the co-authors to Chapter 5 are as follows:

- Profs. Justin Dauwels, Yu Jiang and Francisco C. Pereira guided the work, reviewed it and confirmed the evaluation of its results.
- Kelvin Lee and Prof. Yu Jiang formulated the optimization scheme.
- Kelvin Lee implemented the optimization scheme, conducted the optimization experiments and wrote the corresponding results.
- I designed the goals, research questions, manner of experimentation and choice of Quantile Regression models.
- I retrieved the data, analyzed the results of data-driven demand prediction and derived the main findings of the research.

Chapter 6 is published as Peled, I., Lee, K., Jiang, Y., Dauwels, J., Pereira, F. C. (2021). On the quality requirements of demand prediction for dynamic public transport. *Communications in Transportation Research*, Vol. 1. DOI: 10.1016/j.commtr.2021.100008

The contributions of the co-authors to Chapter 6 are as follows:

- Profs. Justin Dauwels, Yu Jiang and Francisco C. Pereira guided the work, reviewed it and confirmed the evaluation of its results.
- Kelvin Lee and Prof. Yu Jiang formulated the optimization scheme.
- Kelvin Lee implemented the optimization scheme, conducted the optimization experiments and wrote the corresponding results.
- I designed the study and the manner of experimentation.
- I collected the Rejsekort data through communication with several providers, carried out the experiments and analyzed their results.
- I wrote the entire manuscript, except for the subsections on optimization.

Appendix B is published as Peled, I., Pereira, F. C., Winther, O. (2018). Traffic Prediction with Convolutional Long Short-Term Memory. *hEART 2018: 7th Symposium of the European Association for Research in Transportation*.

The contributions of the co-authors to Appendix B are as follows:

- Profs. Ole Winther and Francisco C. Pereira guided the work, reviewed it and confirmed the evaluation of its results.

- I designed and conducted the entire work, including the methodology, models and experiments.
- I built the Deep Neural Network models, experimented with them and analysed the results.

4-Dec-2021

.....

Date

ITU NTU NTU NTU NTU NTU NTU NTU
NTU NTU NTU NTU NTU NTU NTU NTU
Inon Peled
ITU NTU NTU NTU NTU NTU NTU NTU

.....

Inon Peled

Acknowledgements

I would first like to express my deep gratitude to my main supervisor, Prof. Francisco Câmara Pereira. In February 2017, I started working as a Research Assistant in his group, Machine Learning for Smart Mobility (MLSM) in the Technical University of Denmark. A few months later, Francisco advised me to pursue a PhD, and I am very glad I followed his advice. Our mutual work in the ensuing three years was enjoyable and enlightening, and this thesis was facilitated by his guidance and genuine care.

I would also like to thank my co-supervisor, Assoc. Prof. Justin Dauwels, for the time we spent together in Nanyang Technological University in Singapore. It was particularly rewarding to get to know about his rich academic career path and the lessons learnt from it.

I am also thankful to Assoc. Prof. Filipe Rodrigues of the MLSM group, with whom I collaborated on multiple works. Though constantly busy with other research and collaborations, Filipe was always available to offer ideas, advice and friendship.

An especially warm Thank You goes to my family: father Moti, mother Riki, brother Assaf and sister Noa, for their love and encouragement throughout my PhD journey.

Finally, I would like to thank the following for their funding to the research in this thesis: the Technical University of Denmark; Otto Mønstedts Fond; the Urban Innovative Actions (LINC project); and the People Programme (Marie Skłodowska-Curie Actions) of the European Union's Horizon 2020 Research and Innovation Programme.

In order to make progress, one must leave the door to the unknown ajar

Richard Feynman, *The Pleasure of Finding Things Out*

Summary

The goal of the thesis is to develop and study effective modeling methods for Transportation under uncertainty scenarios. This is motivated by both the prevalence of uncertainty in Transportation and the widespread use of Transportation models in practice, e.g., for traffic management, planning of mobility services and operation of Public Transport. We approach this goal through Machine Learning, namely, our proposed methods extract patterns from data and leverage them for better modeling.

The first uncertainty scenario we deal with concerns road incidents. As each incident induces a unique and uncertain change in the correlation structure of traffic variables, prediction models cannot realistically be prepared in advance for every possible incident. We thus devise QTIP: an online framework that uses distress signals from affected vehicles for ad-hoc simulations and corresponding model adaptation, which we experiment in a case study of a Danish highway. The results suggest that QTIP provides a feasible opportunity for mitigating the long-standing problem of instantaneous model adaptation upon incidents.

Next, we deal with modeling of mobility demand when its actual value is uncertain, as it is only partially observed. A common reason for this is the use of data from mobility services, whereby demand is observed through limited service supply and is thus “censored”. First, we propose a novel regression method that combines flexible Gaussian Processes with a censored likelihood, and we experiment it on real-world data from bike-sharing and taxi services. The results emphasize that censorship can and should be considered for more accurate modeling of mobility demand. Second, we propose Censored Quantile Regression Neural Networks as an alternative non-parametric method for modeling mobility demand. We also apply this method to real-world data from bike-sharing and shared Electric Vehicles, and the results show that it can outperform both censorship-unaware Neural Networks and censored parametric models. To the best of our knowledge, no previous works have applied Censored Quantile Regression Neural Networks in the Transportation domain.

Finally, we study the impact of travel demand uncertainty on the performance of demand-responsive Transport services. First, we devise a framework for predictive optimization, which estimates marginal distributions of travel demand between Origin-Destination pairs, joins them through a copula, and uses the joint distribution for stochastic route optimization. We experiment the framework in a case study of on-campus autonomous mobility, where demand is observed via WiFi-probed crowd movements, and obtain that the framework can outperform conventional optimization techniques that do not leverage the full predictive distribution. Second, we study the impact of prediction inaccuracy via a more

general approach, which does not involve specific prediction models, by sampling prediction errors from various distributions. We apply this approach to a case study of demand-responsive Public Transport in the Copenhagen metropolitan area, through which we quantify the relationship between prediction errors and subsequent performance of dynamic routing.

In conclusion, this thesis offers several useful findings for Transportation practice and theory. We find that recent technological advances can alleviate the degradation of data-driven prediction models under road incidents, for which we offer a dedicated framework. We also advise to explicitly model the inherent censorship in Transportation demand, for which we offer two non-parametric alternatives. For dynamic operation of shared mobility services, we demonstrate the benefits of preserving a full uncertainty structure of demand, and we also quantify the relationship between predictive quality and subsequent service performance.

To my dear father, mother, brother and sister.

Contents

Summary (English)	i
Resumé (Dansk)	iii
Preface	v
Acknowledgements	vi
1 Introduction	1
1.1 Motivation	1
1.2 Contributions	4
1.3 Thesis Structure	5
2 Simulation-based Model Adaptation during Incidents	6
2.1 Introduction	7
2.1.1 The Challenge: Just-in-Time Model Adaptation	7
2.1.2 Our Contributions	8
2.1.3 Paper Organization	9
2.2 Current Solutions, Gaps and Opportunities	9
2.2.1 Current Methods for Traffic Prediction Under Atypical Conditions	10
2.2.2 Studying the Effects of Road Incidents through Simulations	11
2.2.3 Motivation for Incorporating Simulations in Real-Time Incident Modeling	11
2.2.4 Real-Time Incident Data for Online Modeling	12
2.3 The QTIP Generic Framework	13
2.3.1 Overall Framework	13
2.3.2 Advantages of QTIP over Existing Solutions	14
2.3.3 Model Selection	15
2.4 Case Study for Evaluating QTIP	17
2.4.1 Overall Design of Experiments	17
2.4.2 The Road Network	19
2.4.3 Data for Calibration	19
2.4.4 Construction and Calibration of Simulated Environment .	21
2.4.5 Experiments	23
2.5 Results	27
2.5.1 Real-Time Performance of QTIP	27
2.5.2 Comparison of Predictive Performance	28
2.5.3 Closer Look at Several Representative Incident Scenarios	29
2.5.4 Model for Ordinary Conditions Degrades under Incidents	30
2.5.5 Improvement of Predictive Performance with QTIP	32
2.5.6 Transfer Learning	34
2.6 Summary of Key Points and Future Work	35
2.6.1 Findings and Implications	38
2.6.2 Limitations	39

2.6.3 Future Work 39

3 Modeling Latent Mobility Demand via Censored Gaussian Processes 41

3.1 Introduction 41

3.2 Review of Related Work 42

3.2.1 Demand Forecasting for Shared Mobility 42

3.2.2 Censored Modeling 43

3.2.3 Common Approaches to Handling Demand Censorship . . 44

3.3 Methodology 45

3.3.1 Tobit Likelihood 45

3.3.2 Gaussian Processes 46

3.3.3 Kernels 48

3.3.4 Gaussian Process with Censored Likelihood 49

3.4 Experiments 51

3.4.1 Models 52

3.4.2 Synthetic Dataset 52

3.4.3 Real-World Dataset 1: Motorcycle Accident 54

3.4.4 Real-World Dataset 2: Bike-Sharing Demand and Supply 57

3.4.5 Real-World Dataset 3: Taxi Demand and Supply 62

3.5 Summary and Future Work 66

3.6 Supplementary Results 67

3.6.1 Matrix-based Imputation 67

3.6.2 Additional Numeric Results 71

4 Modeling Latent Mobility Demand via Quantile Regression Neural Networks 74

4.1 Introduction 74

4.2 Literature Review 75

4.3 Methodology 78

4.4 Demonstrating the Advantages of Censored Quantile Regression . 80

4.4.1 Non-Censored vs. Censored QR 80

4.4.2 Parametric vs. Non-Parametric CQR 83

4.5 Experiments for Estimating Latent Mobility Demand 85

4.5.1 Bike-sharing Data 86

4.5.2 Shared Electric Vehicles Data 88

4.6 Conclusion 90

5 Predictive Optimization Framework for Demand-Responsive Public Transport 91

5.1 Introduction 92

5.1.1 Overview of Our Solution Framework 92

5.1.2 Summary of Contributions 94

5.1.3 Paper Structure 94

5.2 Literature Review 95

5.2.1 Demand Prediction with Uncertainty 95

5.2.2 Transit Network Design and Frequency Setting Problem (TNDFS) 96

5.2.3	Combining Demand Prediction with Supply Optimization	99
5.3	Demand Prediction through Quantile Regression	101
5.3.1	Data for Demand Estimation	101
5.3.2	Quantile Regression	102
5.3.3	Independent Models	104
5.3.4	Modeling Spatio-Temporal Dependencies	109
5.3.5	Gradient Boosting on Multiple Time Series	112
5.3.6	Conclusions on Demand Prediction with Quantile Regression	112
5.3.7	Online Demand Prediction	113
5.4	Supply Optimization	113
5.4.1	Sampling via Gaussian Copula	113
5.4.2	TRNDP as a minimum cost flow problem	116
5.5	Case Study for Stochastic Optimization	120
5.5.1	Optimization with Ground Truth vs. Predictions	122
5.5.2	Comparison to Conventional Optimization Methods	123
5.5.3	Online Supply Optimization	124
5.6	Conclusion	124
5.6.1	Future Work	125
6	On the Quality Requirements of Demand Prediction for Dynamic Public Transport	127
6.1	Introduction	127
6.2	Literature Review	129
6.2.1	Uncertainty in Modeling and the Normality Assumption	129
6.2.2	Public Transport Optimization	130
6.3	Experiments	131
6.3.1	Data	132
6.3.2	Generation of Noisy Demand Predictions	132
6.3.3	Fleet Optimization	133
6.4	Results	136
6.5	Discussion	141
6.6	Conclusion	142
6.7	Supplementary Results	143
7	Conclusion and Future Work	148
7.1	Findings and Key Messages	148
7.2	Future Work	150
	References	152
A	Model-Based Machine Learning for Transportation	168
A.1	Introduction	168
A.2	Case Study 1: Taxi Demand in New York City	170
A.2.1	Initial Probabilistic Model: Linear Regression	170
A.2.2	Key Components of MBML	171
A.2.3	Inference	174
A.2.4	Model Improvements	176

A.3	Case Study 2: Travel Mode Choices	181
A.3.1	Improvement: Hierarchical Modeling	183
A.4	Case Study 3: Freeway Occupancy in San Francisco	184
A.4.1	Autoregressive Model	184
A.4.2	State-Space Model	185
A.4.3	Linear Dynamical Systems	186
A.4.4	Common Enhancements to LDS	187
A.4.5	Non-Linear Variations on LDS	189
A.5	Case Study 4: Incident Duration Prediction	191
A.5.1	Preprocessing	192
A.5.2	Bag-of-Words Encoding	193
A.5.3	Latent Dirichlet Allocation (LDA)	193
A.6	Summary	196
A.6.1	Further Reading	196
B	Traffic Prediction with Convolutional Long Short-Term Mem-	
	ory	198
B.1	Introduction	198
B.2	Data Description and Preparation	199
B.2.1	Data Cleaning	200
B.2.2	From Output to Prediction	201
B.3	Baseline	201
B.4	Deep Neural Networks	204
B.4.1	Selection of Hyper-Parameters	205
B.4.2	Neural Networks with FC-LSTM	205
B.4.3	LSTM with Separated Input	205
B.4.4	LSTM with Combined Input	206
B.4.5	Mixture of Experts	207
B.4.6	Dropout and Regularization	207
B.4.7	LSTM with Entire Place Group Input	208
B.4.8	Convolutional LSTM	208
B.4.9	Summary of Results	209
B.5	Conclusions and Future Work	210

List of Tables

2.1	Vehicle counts	20
2.2	Calibrated parameters	22
2.3	Model performance	28
2.4	$M_{ordinary}$ performance	31
3.1	Weather features for bike-sharing data	60
3.2	Performance incl. NCGP-I for entire data of super-hub 1	70
3.3	Model Performance, with NCGP-I, for Entire Taxi Dataset	70
3.4	Model Performance for Super-hub 1	72

3.5	Model Performance for Super-hub 2	72
3.6	Model Performance for Super-hub 3	72
3.7	Model Performance in RandDropoff Experiments	73
4.1	Percent of zero conditional quantiles in each synthetic dataset . .	82
4.2	Predictive quality of conditional quantiles of synthetic y^*	84
4.3	Non-parametric QR vs. parametric Tobit on synthetic datasets .	85
4.4	Results of experiments with shared EV data	90
5.1	Predictive performance of Quantile Regression models	104
5.2	Notations used in the minimum cost flow formulation.	118
5.3	Commonly recurring solutions in the case study results	121
5.4	Hourly optimal solutions	122
5.5	Hourly optimal solutions using different optimization strategies .	123
6.1	Skew and kurtosis of error distributions	134
6.2	Notation for optimization formulation	136
6.3	No. dynamically and statically routed buses	137
6.4	Mean travel time	138
6.5	Minimum yearly economic gains	140
6.6	Time lost	144
6.7	Time relatively lost	145
6.8	Time gained	146
6.9	Time relatively gained	147
A.1	Probability theory concepts used in this chapter	169
A.2	Summary statistics for prediction errors	180
A.3	Predictive performance of M_1^{Pois} vs. M_1	180
B.1	Performance of candidate baseline models for speed	203
B.2	Performance of candidate baseline models for flow	203
B.3	Linear Regression performance given both speeds and flows . . .	204
B.4	Speed prediction quality for all models	209
B.5	Flow prediction quality for all models	210

List of Figures

2.1	QTIP framework	13
2.2	Different incident scenarios	18
2.3	Studied links	19
2.4	Hillerød Motorway	20
2.5	Calibration results	23
2.6	Road blockage experiments	24
2.7	DNN Structure in experiments	26
2.8	Representative results	30

2.9	$M_{ordinary}$ performance	31
2.10	$M_{ordinary}$ degradation on average time series	32
2.11	$M_{abnormal}$ vs. $M_{ordinary}$	33
2.12	RMSE as no. simulations increases, lanes known	36
2.13	RMSE as no. simulations increases, lanes unknown	37
3.1	Experiments with synthetic data	53
3.2	NCGP fit on motorcycle accident data	54
3.3	NCGP-A fit on motorcycle accident data	55
3.4	CGP fit on motorcycle accident data	55
3.5	Bike-sharing hubs and super-hubs	57
3.6	Super-hub 3 data with 50% Censorship Intensity.	59
3.7	Model performance on bike-sharing data	61
3.8	RandDropoff censorship	63
3.9	Example of RandDropoff censorship	64
3.10	Model performance in RandDropoff experiments	65
3.11	CGP vs. NCGP-I on the synthetic dataset	68
3.12	NCGP-I fit on the motorcycle dataset.	69
3.13	Performance incl. NCGP-I for bike-sharing data	69
4.1	Our Censored Quantile Regression Neural Network architecture.	78
4.2	Tilted Loss.	79
4.3	Synthetic y^* distributions	81
4.4	Conditional quantiles of synthetic y^*	81
4.5	Bikes-haring hubs and aggregated superhubs	86
4.6	Bike-sharing results on all test set	87
4.7	Bike-sharing results on non-censored test observations	88
4.8	Shared EV data and $\alpha = 40\%$ fleet reduction.	89
5.1	Online predictive optimization framework	93
5.2	Data in our case study	101
5.3	Example output of a Quantile Regression model	103
5.4	DNN^{Ind}	107
5.5	Bayesian Optimization of learning rate (LR) of DNN^{Ind}	108
5.6	Gradient Boosting model	109
5.7	Optimal learning rate distribution over all OD pairs	109
5.8	Pearson correlation between OD pairs	110
5.9	Multivariate deep Quantile Regression models	111
5.10	Minimum cost flow formulation	117
5.11	Nodes in the case study	121
5.12	Prediction vs. optimization quality for best performing models	123
6.1	Public Transport stations	132
6.2	Probability Density Functions of the error distributions	134
6.3	Optimization performance	138
6.4	Average minutes lost with noisy vs. perfect predictions	139
6.5	Average minutes gained with dynamic vs. static routing	140
6.6	Minimum time gained relatively	141

A.1	Generative process for linear regression model M_1	172
A.2	Probabilistic Graphical Models for case study 1	173
A.3	Approximation of intractable posterior distribution	175
A.4	Generative process for heteroscedastic model M_1^{Het}	177
A.5	Homoscedastic model vs. heteroscedastic model	179
A.6	Generative process for M_1^{Pois}	179
A.7	Classification model	182
A.8	Probabilistic Graphical Models for case study 2	184
A.9	Generative process for model M_2^{Hier2}	185
A.10	PGM for model M_2^{Hier2}	185
A.11	Generative process for $AR(k)$ model	187
A.12	State-Space Model (SSM) and Linear Dynamical System (LDS)	187
A.13	PGM for LDS with imputation and forecasting	188
A.14	Imputation with M_3^{LDS}	189
A.15	LDS with either external data or regimes	190
A.16	PGM for Hidden Markov Model (HMM)	191
A.17	Generative process for Hidden Markov Model	191
A.18	Prediction of incident duration in minutes	192
A.19	Topic modeling of incident reports	193
A.20	PGM for M_{LDA}	195
A.21	Generative process for M_{LDA}	195
B.1	Mean speed in each unique place in the dataset	200
B.2	RMSE of the three baseline candidate models	203
B.3	Neural Network architectures in our experiments.	204
B.4	Exhaustive search for best hyper-parameters	206
B.5	RMSE for all models	207

Introduction

This Chapter describes the unifying theme of all works in this thesis: Machine Learning (ML) methods for Transportation modeling under uncertain conditions. After a brief overview of ML, I discuss the need to account for uncertainty in modeling. I then review ML methods for modeling with uncertainty, both in general and particularly in the Transportation domain. Finally, I explain how the works in the following Chapters fit into the aforementioned theme, and in particular, which knowledge gaps they address and how they do so. In conclusion, I list my main contributions and describe the thesis structure.

1.1 Motivation

Traffic Management Centers and Intelligent Transportation Systems rely on predictive models of short-term and long-term traffic (Nosratabadi et al., 2019). These models typically employ either classic Transport Engineering techniques or modern Machine Learning methods. The classic techniques are oriented toward incorporating pre-specified patterns, such as Origin-Destination matrices for assignment of flows to traffic networks (Antoniou et al., 2016) and specifications of driver behavior for micro-simulations of road conditions (Balakrishna et al., 2007). These techniques thus represent a “white box” approach to modeling, which enables close access to the assumed dynamics of Transportation. In contrast, modern Machine Learning (ML) methods rely on algorithms to automatically detect patterns in data (Bishop, 2006). ML can thus be generally viewed as a “black box” approach to modeling, in which some control of specific dynamics is traded for greater freedom of pattern recognition.

ML methods originated in statistical methods for describing and quantifying structural patterns in data (Finlay and Agresti, 1986), and together with the growth of computational power, progressed rapidly since the 1950s (Nilsson, 1965). ML methods are mainly used in regression (for learning continuous functions), classification (for learning discrete functions) and clustering (for grouping by similarity) (Harrington, 2012). Some ML methods are parametric, i.e., assume a particular functional relationship between input and output, while others follow a more flexible, non-parametric form. Some are model-based (Appendix A), whereby a generative process of data is first defined, while others are data-driven, such that the data itself guides the learning process. ML methods can also be categorized as either supervised, semi-supervised or unsupervised,

depending on whether they are trained with fully labeled, partially labeled or unlabeled inputs, respectively, where labels indicate intended output.

Nowadays, ML methods are used across multiple research disciplines and areas of application, including Medicine (Rajkomar et al., 2019), Economics (Athey, 2018), Manufacturing (Jordan and Mitchell, 2015), Robotics (Murphy, 2019) and Transportation (Antoniou et al., 2018, Nagy and Simon, 2018). The development of modern, Intelligent Transportation Systems employs ML in analyzing large data volumes from road sensors, traffic simulators, travel surveys, smart-phone usage, and more recently, Vehicle-to-everything (V2X) communications (Bhavsar et al., 2017). Traffic practitioners use ML algorithms for real-time incident management (Li et al., 2018). Autonomous vehicles rely heavily on ML to comprehend their environment and choose subsequent actions (Koopman and Wagner, 2017). International freight shipping uses ML algorithms to choose transportation modes and routes (Barua et al., 2020). Public Transport, including trains, buses and taxis, uses ML predictions of passenger loads for service optimization and dynamic pricing (Branda et al., 2020). Airports apply ML in capacity estimation and early detection of flight delays (Gui et al., 2019). Shared mobility services, such as ride-hailing and bike-sharing, rebalance their fleets based on ML estimates of travel demand (Wen et al., 2017).

These and many other examples have a common property: the existence of *uncertainty* in the modeled phenomena, whether because of incomplete knowledge (epistemic uncertainty) or innate non-determinism (aleatoric uncertainty) (Beaudrie et al., 2016). For instance, when a nonrecurrent event draws in big crowds, Public Transport usage may surge or drop by an uncertain magnitude (Rodrigues et al., 2019). Another instance of uncertainty in Transportation is the high variability in the duration and impact of different road incidents (Pereira et al., 2013). Weather conditions also pose uncertain circumstances for effective traffic management (Maze et al., 2006). Transportation data itself may contain an uncertain amount of noise, e.g., because of omission (van Lint et al., 2005), inaccuracy (Bernas et al., 2018) or inherent bias (Albiński et al., 2018).

Models of reality, including Transportation models, must therefore account for uncertainty, and ML methods handle this subject in various manners¹. In parametric modeling, uncertainty is commonly quantified through the calculation of confidence intervals for the predictive mean and prediction intervals for realizations from the predictive distribution (Heskes, 1997). These intervals, in turn, are calculated through various methods, such as analytic derivations of the predictive variance (Solomatine and Shrestha, 2009) and model averaging via input sampling (Efron, 1982). In some situations, however, it is advantageous to estimate a full predictive distribution, e.g., to account for skew (van Lint

¹The term “Uncertainty Analysis” is sometimes used in this context, but more often refers to design of Engineering experiments (Coleman and Steele, 2018).

et al., 2008) or heteroscedasticity (Pereira et al., 2014).

Several non-parametric ML methods are designed to intrinsically estimate a full predictive distribution. Quantile Regression (Koenker and Bassett, 1978) approximates the predictive distribution by modeling multiple quantiles of its Cumulative Distribution Function. Probabilistic Graphical Models (Koller and Friedman, 2009) explicitly encode a joint probability distribution of the data variables, which can then be inferred through various exact or approximate methods. Gaussian Processes (Rasmussen, 2003) place a multivariate Gaussian prior over functions, where the Gaussian covariance matrix expresses similarity between the input variables. Several Neural Network formulations have been designed to learn distributions, e.g., ensemble methods for confidence intervals (van Hinsbergen et al., 2009), Variational Autoencoders and Generative Adversarial Networks for generative modeling (Creswell et al., 2018), and Bayesian Neural Networks for preserving uncertainty in model weights (Blundell et al., 2015).

In the Transportation domain particularly, multiple works have applied ML methods for modeling under uncertainty, as also reviewed in the following surveys. De Jong et al. (2007) review works that account for uncertainty in forecasting flows, costs, revenues, travel times and other traffic aspects. Rasouli and Timmermans (2012) review works that account for uncertainty in prediction of mobility demand. Narayanan et al. (2020) review works on shared autonomous vehicles and methods for estimating uncertainty in future demand for this new mobility mode. Corman and Meng (2015) review models for online railway traffic management and conclude that future models should better treat uncertainty in this operational area. Ibarra-Rojas et al. (2015) review works on planning, operation, and control of bus transport systems under uncertain arrival times, boarding/alighting times and travel times. Schlögl and Stütz (2019) review spatio-temporal uncertainty in datasets, which are commonly used in road safety analysis, and suggest methods for handling this uncertainty.

Nevertheless, even after decades of research into ML methods for Transportation under uncertainty, some knowledge gaps still remain, and this thesis addresses several of them. First, in Chapter 2, we recognize that each road incident induces a unique and uncertain change in the correlation structure of traffic variables, which in turn, can significantly deteriorate the predictive quality of data-driven traffic models. We further note that traffic simulations can generate incident data for additional model training, yet the space of all possible incidents is too vast to fully simulate in advance. Consequently, we devise an online framework that executes incident simulations ad-hoc and uses their output to adapt data-driven traffic prediction models. The framework leverages distress signals from affected vehicles, and we demonstrate its real-time capabilities in a case study of a Danish highway.

Next, we proceed to address issues in modeling mobility demand under data censorship, as follows. In Chapter 3, we note that mobility demand is often observed through usage of mobility services, so that data about it is inherently biased (“right-censored”), as supply limits can obscure some of the actual demand. We then note that many existing studies treat censorship through omission and imputation, rather than explicitly model it to better estimate latent mobility demand. Consequently, we propose a novel and flexible censored regression method based on Gaussian Processes, which we demonstrate on both synthetic data and real-world data from bike-sharing and taxi services. As Gaussian Processes face some limitations when scaling to big datasets, we next propose in Chapter 4 to model latent mobility demand via Censored Quantile Regression Neural Networks. To demonstrate the advantages of this alternative non-parametric method in reconstructing latent information, we also apply it to several case studies of censorship in synthetic and real-world shared mobility data. To the best of our knowledge, our work is the first to apply this censored modeling method in the Transportation domain.

Finally, we study the effects of travel demand uncertainty on subsequent performance of demand-responsive Transport services. In Chapter 5, we propose to model a full uncertainty structure of travel demand, rather than reduce it to point estimates as more common. For this, we propose a framework that constructs marginal distributions of travel demand between Origin-Destination pairs, combines the marginals into a joint distribution and uses it to optimize a shared Transport service. We experiment the framework in a case study of autonomous mobility in DTU Lyngby campus, wherein travel demand is measured by crowd movements between buildings, as observed through WiFi probing. In Chapter 6, we then study the impact of predictive uncertainty on demand-responsive Public Transport via a case study on smart card usage in buses, trains and metro in the Copenhagen metropolitan area. Rather than apply specific prediction models as in other studies, we simulate prediction errors by sampling from various distributions, and in particular, measure the impact of non-normally distributed residuals. Hence similarly to our generic framework in Chapter 2, Chapter 6 too generalizes beyond specific ML methods.

1.2 Contributions

In summary, following are my personal contributions in this thesis.

- A simulation-based framework for real-time adaptation of traffic prediction models upon road incidents, which combines two traditionally separate modeling approaches: “black box” Machine Learning and “white box” Transportation Engineering.

- Application of a Gaussian Process-based censored regression method to a case study of taxi demand in New York City, along with formulation of a stochastic, supply-based censorship scheme.
- A study on the effectiveness of Censored Quantile Regression Neural Networks in modeling latent mobility demand, which is apparently the first work to apply this method in the Transportation domain.
- A method for constructing marginal distributions of travel demand, and a case study for using it in a framework for predictive optimization of on-campus autonomous mobility.
- A study on the impact of predictive inaccuracy on subsequent performance of demand-responsive Public Transport, and in particular, the impact of violating a common assumption about normally distributed residuals.

1.3 Thesis Structure

The rest of this thesis is organized as follows. Chapter 2 studies a simulation-based framework for quick adaptation of data-driven prediction models when road incidents occur, at which time the uncertainty structure of traffic suddenly changes. The next two Chapters deal with non-parametric methods for estimating the latent distribution of demand for shared mobility services, where the actual demand is uncertain as it is observed through limited vehicle supply. Chapter 3 does so through Censored Gaussian Processes, whereas Chapter 4 uses Censored Quantile Regression Neural Networks.

The next two Chapters study the relationship between uncertainty in travel demand and performance of demand-responsive Transport services. Chapter 5 proposes a framework for operating such services while estimating a full distribution of demand, and applies it to a case study of on-campus autonomous mobility. Chapter 6 then studies the impact of errors in demand predictions on subsequent performance of demand-responsive Public Transport. Finally, Chapter 7 summarizes the main findings in the thesis and concludes with plans for future work.

While the above-mentioned Chapters contain my main works on modeling Transportation under uncertainty, the following Appendices include two additionally related works, where I am the main author. Appendix A (co-authored with Assoc. Prof. Filipe Rodrigues and Prof. Francisco C. Pereira) reviews Model-Based Machine Learning methods, which yield a full predictive distribution, and applies them in several Transportation case studies. Appendix B (co-authored with Profs. Ole Winther and Francisco C. Pereira) studies the effectiveness of Convolutional Long Short-Term Memory for leveraging spatio-temporal correlations in traffic modeling.

Simulation-based Model Adaptation during Incidents

This Chapter is based on a mutual work with several past and present MLSM members: Raghuveer Kamalakar, whose MSc thesis formed a basis for the work; Assoc. Prof. Carlos M. Lima Azevedo; and my PhD supervisor, Prof. Francisco C. Pereira. I am the main author, and the work was published in May 2020 in Journal of Simulation (Peled et al., 2020).

The work is motivated by both a problem with data-driven traffic prediction models and an emerging technological opportunity. Data-driven models are typically trained with large datasets that pertain mostly to ordinary road conditions, and so they usually perform well under incident-free conditions. Unfortunately, these models often fail when traffic suffers a sudden and significant disruption, such as a road incident – just when reliable predictions are most needed for effective traffic management. Nowadays, however, an increasing percentage of vehicles are equipped with systems that emit immediate distress signals, which offer an opportunity for quickly recognizing and responding to road incidents. This motivates us to study the possibility of adapting traffic prediction models as soon as an incident happens.

The outcome of our study is QTIP: a simulation-based framework for quasi-instantaneous adaptation of prediction models upon traffic disruption. Put briefly, QTIP performs real-time simulations of the affected road for multiple possible scenarios, analyzes the results, and suggests a change of prediction model accordingly. It constructs the simulated scenarios per properties of the incident, as conveyed by immediate distress signals from affected vehicles. To empirically evaluate QTIP, we experiment it in a case study of a Danish highway, which demonstrates that QTIP can improve traffic prediction in the first critical minutes of road incidents.

The case study also demonstrates a benefit of QTIP, whereby the simulations do not require online calibration. Rather, they can initially be calibrated offline with accurate road topology, typical driver behavior and other parameters that hold under future incidents. Then when an incident occurs, the remaining parameters need not be calibrated online either – QTIP can cover their range of possible values via multiple real-time simulations, which in turn generate data for effective Machine Learning prediction models.

Within the theme of my thesis, this Chapter thus relates to the epistemic uncertainty brought about by nonrecurrent incidents, and offers a framework for alleviating the typical deterioration of Machine Learning models under such conditions. Before its publication, this work went through several review cycles, which underlined for me the importance of effective scientific writing, whereby the motivation, goals, methodology and contributions of the work are clearly conveyed.

2.1 Introduction

Nonrecurrent traffic disruptions are a major source of travel delays and air pollution in urban environments (Tupper et al., 2012, Vlahogianni et al., 2010). As urban traffic around the world increases constantly, main roads encounter more vehicle breakdowns, crashes, adverse weather, and large public events (Kwon et al., 2006). Consequently, a growing amount of resources is being invested world-wide in the study and treatment of traffic incidents (Bertini et al., 2005, Kong et al., 2013, Mir and Filali, 2016, Wang, 2010).

Prediction models form a key component of traffic incident management for both short-term operations and long-term planning (Ben-Akiva et al., 1998). Nevertheless, research into traffic prediction has concentrated mostly on incident-free conditions (Castro-Neto et al., 2009, Salamanis et al., 2017). In addition, prediction models in practical use often rely on commonly available traffic data streams, as e.g. generated by mobile sensors and on-road cameras (Wu et al., 2012). Unfortunately, such models are slow to adapt to sudden traffic disruptions, during which effective incident treatment is most needed.

2.1.1 The Challenge: Just-in-Time Model Adaptation

Under ordinary conditions, speeds and travel times tend to follow consistent trends, hence real-time predictions can be made to often come close to actual values. However, under nonrecurrent disruptions, the accuracy of real-time predictions can deteriorate greatly, and dedicated methods are needed for increased accuracy (Chung and Recker, 2013). Indeed, immediate adaptation of traffic prediction models to sudden disruptions using real-time data has so far been a largely unsolved problem. Several approaches to online model adaptation have been proposed (Castro-Neto et al., 2009, Ni et al., 2014, Wu et al., 2012), showing that model adaptation is needed to prevent significant deviation of predicted values (e.g., mean speed) from actual measurements. Nevertheless, these existing approaches all assume a time buffer for adaptation, namely, they

yield an adapted model only after collecting online traffic data for a few minutes following an incident.

Nowadays, however, more and more vehicles are being equipped with In-Vehicle Monitoring Systems (IVMS) (EU European Commission, 2015, Viereckl et al., 2016), which communicate real-time distress signals upon vehicle breakdown. IVMS thus offers a two-fold opportunity for online model adaptation: (1) immediate triggering, and (2) additional information about the particular circumstances of the incident. While predicting the occurrence of traffic incidents remains a challenge in itself (Katrakazas et al., 2018), this paper provides empirical evidence for the possibility of quick model adaptation once an incident is known to have occurred.

2.1.2 Our Contributions

The prime contribution of this paper is QTIP: a novel framework for Quick Adaptation of Traffic Model per Incident Parameters. The novelty here lies in the combination of two traditionally separate approaches for traffic modeling, namely: data-driven machine learning and classic transport engineering methods. To realize the benefits of this combination, let us now present the two approaches and their complementary aspects.

Given a modeling problem (e.g., speed prediction in this paper), *data-driven machine learning* uses algorithms to automatically extract useful patterns from corresponding observation data. The data consists of response variables (e.g., speed) and explanatory variables (e.g., time of day and weather). The algorithms themselves may be either parametric or non-parametric, depending on whether or not they assume a particular functional relationship between response and explanatory variables. Machine Learning algorithms thus serve as *black boxes* that take structured data as input and yield trained models as output.¹

On the other hand, *classic transport engineering methods* are more principled and oriented towards incorporation of "behavior" in modeling. For instance, analytical formulations of dynamic traffic assignment (Boyce et al., 2001) use origin-destination matrices and network topology to assign traffic flows on the network, and micro-simulators for traffic modeling rely on detailed specification of driver behavior (e.g., safety distance, braking, lane changing, rubbernecking) and road conditions (e.g. presence of pedestrians, ratio of heavy to light vehicles). Such methods are thus concerned with detailed specification of the problem through *white boxes* which allow close access to underlying dynamics.

¹We are oversimplifying here for the sake of the argument, as there are also "white-box" machine learning approaches, such as Probabilistic Graphical Models (Peled et al., 2019c).

QTIP, the proposed traffic modeling framework, takes advantage of both these approaches in a manner that depends on road conditions. Under normal traffic conditions, QTIP yields a purely data-driven model which is constructed from historical observations of ordinary traffic, as is common practice. For abnormal traffic conditions, however, QTIP generates multiple simulations, which reflect the likely range of specific properties of the road incident, and uses their output as data for fitting a specialized machine learning model for the incident. QTIP thus enables the use of powerful machine learning methods not only under normal and repetitive traffic conditions, but also in the first critical minutes of nonrecurrent incidents.

In conclusion, we hereby enumerate all contributions of this paper.

1. QTIP: a solution methodology for quasi-instantaneous adaptation of traffic prediction models, based on a novel combination of traditionally separate modeling approaches: "black-box" machine learning and "white-box" transport engineering methods.
2. Empirical case study for a major motorway in Denmark measurably demonstrates:
 - (a) The uniqueness of each incident.
 - (b) The degradation of prediction models under road incidents.
 - (c) The potential of QTIP in mitigating this degradation.
3. Code for generating and analyzing simulated scenarios given incident information is shared in https://github.com/inon-peled/qtip_code_pub.

2.1.3 Paper Organization

The rest of this paper is organized as follows. Section 2.2 provides a literary review of current solutions for real-time incident modeling. Section 2.3 then describes the QTIP framework, and Section 2.4 defines a case study for evaluating QTIP. Section 2.5 provides the results of the case study, and Section 2.6 concludes with a summary of our findings.

2.2 Current Solutions, Gaps and Opportunities

In this section, we first review current solutions for modeling atypical traffic conditions, and the usefulness of simulations for studying incident conditions.

Then, we motivate the necessity of online incident simulations for timely adaptation of data-driven prediction models. Finally, we describe a newly emerging source of real-time incident information, which is highly useful for just-in-time modeling.

2.2.1 Current Methods for Traffic Prediction Under Atypical Conditions

Accurate short-term traffic prediction is essential for proactive applications of Intelligent Transport Systems (ITS), such as Advanced Traveller Information Systems, Dynamic Route Guidance, and Traffic Control (Guo et al., 2012b). Nonrecurrent road incidents disrupt normal traffic patterns, and so increase uncertainty about the near future state of traffic, which thus becomes more challenging to predict. Nevertheless, traffic prediction literature has dealt much more with normal conditions than with incident conditions (Castro-Neto et al., 2009, Salamanis et al., 2017).

The few studies which do cater for both ordinary and incident conditions often use exclusively the data-driven approach (Zhang et al., 2011). Salamanis et al. (Salamanis et al., 2017) analyze 10 years of traffic flow and incident data under the assumption that incidents can be categorized into easily identifiable classes, and cluster the data accordingly. Thereafter, they fit k -Nearest Neighbors (kNN), Support Vector Regression (SVR), and Autoregressive Integrated Moving Average (ARIMA) models to each cluster, and conclude that 5 min prediction accuracy improves when selecting a best performing model per traffic in the preceding hour. In (Guo et al., 2012b, 2014, 2010), Guo et al. successively improve a set of tools for traffic prediction under normal and incident conditions. Their data-driven framework boosts performance through data smoothing and error feedback, and they consistently obtain that under abnormal traffic conditions, kNN-based methods outperform other prediction models, such as SVR, Artificial Neural Network (ANN), and Gaussian Processes (GP). The data-driven models by Salamanis et al. and Guo et al. are thus trained offline, and do not use real-time information from incidents.

Real-time model adaptation to abrupt changes in traffic conditions has been an active research subject in recent years. Wu et al. (Wu et al., 2012) develop an Online Boosting Non-Parametric Regression (OBNR) model for transitioning between normal and incident conditions. OBNR is thus non-parametric, and relies on historical records for online adaptation. Castro-Neto et al. (Castro-Neto et al., 2009) show that under atypical conditions, Online Support Vector Regression (OL-SVR) outperforms Gaussian Maximum Likelihood, Holt exponential smoothing, and ANN. The prediction quality of OL-SVR gradually improves as data from Vehicle Detector Stations accumulates over time. Ni et al. (Ni et al.,

2014) offer social network Twitter as a source of real-time information, which can improve prediction accuracy for traffic around large crowd events. Nevertheless, incorporating social data into real-time traffic analysis incurs some practical challenges, such as the need to collect, clean and fuse social data from multiple sources (Zheng et al., 2016).

2.2.2 Studying the Effects of Road Incidents through Simulations

Simulations are a widely used tool for studying both the short-term and long-term effects of road incidents (Owens et al., 2010). For short-term effects, Henchey et al. (Henchey et al., 2014) use simulations to study emergency response, while Hawas et al. (Hawas, 2007) replicate real-world accidents to analyze car-following models. For long-term effects, Wirtz et al. (Wirtz et al., 2005) study micro-simulations of incidents for proactive planning, while Baykal-Gursoy et al. (Baykal-Gursoy et al., 2006) use micro-simulations to compare strategies of traffic incident management, and Dia et al. (Dia et al., 2006) simulate an Australian highway to measure the socio-economic impacts of incidents.

These former studies, as opposed to this paper, do not deal with online interfacing of simulations with machine learning for model adaptation. Standing in contrast in this respect is a line of works by Ben Akiva et al. (Ben-Akiva et al., 1998, 2002, 2010, 1994, Lu et al., 2015) over DynaMIT: a framework for online traffic modeling through real-time simulations. However, DynaMIT is concerned with modeling traffic conditions on a network-wide level (e.g., the overall state of congestion in a city) in the context of interactions between transport demand and supply. Conversely, this paper focuses on incident conditions in a purposely constrained environment and assumes only limited prior knowledge of transport demand. Our proposed solution framework is thus complementary to the overall vision of DynaMIT, and we indeed suggest to incorporate it as a component in DynaMIT (Section 2.6.3).

2.2.3 Motivation for Incorporating Simulations in Real-Time Incident Modeling

Real-time data-driven prediction models take advantage of the relative stability of conditions over short, consecutive intervals. For example, explanatory variables such as time-of-day, day-of-week, effects of seasonal trends, and weather typically change very little over 5 consecutive minutes. Hence the closer a data-driven model is to real-time resolution, the lower is its necessary complexity (e.g., non-linearity, more explanatory variables), as the effects from trends and context are already incorporated in the current time window.

In contrast, when an incident happens, the correlation structure between response and explanatory variables changes abruptly, in a manner which is unique to the incident characteristics. For example, when an incident occurs, the mean speed in the current time interval may significantly change its pattern of dependency on speeds in recent time intervals, in the affected link and in its neighboring links. As such, there are advantages to treating incidents separately from other atypical conditions, through a dedicated prediction modeling framework.

On one hand then, data-driven prediction models tend to break under the sudden change of correlation structure brought about by an incident. On the other hand, such models could perform well if their input data adequately pertained to the given incident. Unfortunately, historical road incident data is often deficient or altogether missing (Kumar and Toshniwal, 2015).

At first sight, it may seem worthwhile to try and overcome this lack of data by pre-generating sufficiently many incident simulations offline. However, such attempts would in fact be impractical, because every incident involves too many varying parameters (e.g., location, road attributes, weather conditions, lane occupancy levels, vehicles involved, proximity to control systems, etc.). Consequently, only a very restricted subset of all possible combinations of incident parameters can be covered offline, and a data-driven model trained in such manner will necessarily underperform on some out-of-sample scenarios.

It follows that to successfully take advantage of data-driven methods, a framework for prediction modeling under real-time incidents must generate online data which corresponds to the unique parameters of each incident occurrence. This motivates us to consider real-time simulations as a means of generating such data online. For these online simulations, we next describe a useful and globally emerging source of real-time incident information.

2.2.4 Real-Time Incident Data for Online Modeling

As the vision of always-connected cars (V2X) progresses worldwide (Siegel et al., 2018), active In-Vehicle Monitor Systems (IVMS) are becoming increasingly prevalent nowadays (Brandl, 2016, Viereckl et al., 2016). In fact, certain IVMS systems are now mandatory by law, e.g. the European Union now mandates that the eCall system be installed in every new vehicle (EU European Commission, 2015). Real-time signals from IVMS are designed to indicate the status and location of vehicles, and in particular, the occurrence of malfunctions and crashes (Digicore Australia, 2017).

Therefore, IVMS delivers not only immediate indication of incident occurrence, but also rich information about the particular properties of the incident. In the

Figure 2.1: QTIP framework.

next section, we present a framework which employs IVMS signals in real-time simulations for corresponding prediction model adaptation.

2.3 The QTIP Generic Framework

2.3.1 Overall Framework

In this section, we present the framework of QTIP, and illustrate how it differs from current solutions. In the remaining sections thereafter, we evaluate several different instances of this framework against an experimental case study, and show how QTIP can address the first critical minutes of road incidents.

Figure 2.1 summarizes the main components of our QTIP framework. The input to QTIP consists of common data about traffic – e.g. from road sensors, mobile sensors, and weather stations – and incident-specific information from IVMS. QTIP uses both data streams to output a traffic prediction model, as follows.

On one hand, when no incident is known to have occurred, QTIP directly yields model $M_{ordinary}$, which is fit for incident-free conditions. For example, $M_{ordinary}$ can be any desired data-driven model, pre-trained on historical records and consistently updated on recent traffic data.

On the other hand, upon receiving IVMS signals from vehicles involved in a road incident, QTIP yields an adapted traffic prediction model $M_{abnormal}$ by executing multiple simulations. Let us now elaborate on the purpose of these incident simulations and the manner in which they are implemented.

The purpose of the incident simulations is to cover a range of *unobserved* explanatory variables that determine how severely the incident affects its surroundings. In this paper, we use two such variables as example: level of road usage (i.e., "traffic demand") at the moment of the incident, and the precise position of the incident. Each executed simulation thus pertains to a different combination of possible values for the unobserved variables, while also accounting for the *observed* information in the IVMS distress signals, e.g., time of occurrence, number of signals, and general location on the road network.

The simulations are then implemented through two main steps. First, a simulated environment of the affected road (e.g., the motorway in our case study) is constructed and calibrated to resemble its real-world structure. This step requires a dedicated solution component, which we denote in Figure 2.1 on the previous page as Ψ , and which we purposely leave out to future work.

Admittedly, we do not intend to offer here a complete and operational solution, but rather provide a theoretical study of challenges and benefits in extracting value from real-time signals with limited incident information. This study does show that even partial incident information – e.g., one that lacks data about current traffic demand – can still be useful for noticeably improving traffic prediction quality. We also note that pre-calibrated simulation environments can be prepared in advance for roads that are known to be incident-prone, so that QTIP is ready to simulate real-time incidents per their unique characteristics. In fact, the QTIP case study in this paper uses such a pre-calibrated simulation environment for an incident-prone motorway in Denmark.

Once the simulated environment (namely, the affected road) is constructed and calibrated, the second step is to use it for executing the desired simulations. To this end, we use PTV VISSIM as the underlying micro-simulation engine and utilize its Component Object Model (COM)-based Application Programming Interface (API). Through this API, we bootstrap each simulation in real-time per the corresponding variable values – both observed and unobserved – and run all simulations in parallel. QTIP then uses the simulation results to fit and output $M_{abnormal}$, the adapted prediction model.

2.3.2 Advantages of QTIP over Existing Solutions

Let us now highlight several desirable properties of the QTIP framework, which current solutions lack to some extent, as reviewed in Section 2.2. First and foremost, QTIP is designed to readily take advantage of information from the incident itself, as the change in correlation structure between response and explanatory variables is unique to each incident. And so, whereas the prediction

quality of e.g. OL-SVR (Castro-Neto et al., 2009) gradually improves over time, QTIP yields a completely adapted model shortly after incident parameters are known. As we show in Section 2.4, a few incident parameters could indeed be enough for QTIP to yield an effective new model.

Second, QTIP is agnostic to the specific form of $M_{ordinary}$ and $M_{abnormal}$. These models can thus be chosen freely, e.g. as parametric and interpretable models, as we further elaborate in Section 2.3.3. Hence whereas OBNR (Wu et al., 2012) is non-parametric, QTIP easily allows insights into how a prediction model changes when adapted to different traffic conditions. Furthermore, OBNR relies on historical records for online adaptation, whereas QTIP does not require past examples of incidents to yield an adapted model.

In fact, QTIP is also agnostic to the specifics of its input traffic data, which can thus consist of both sensor readings and relevant feeds from social networks, as suggested by Ni et al. (Ni et al., 2014). Nevertheless, while QTIP welcomes such contextual information, its immediate response relies only on signals which originate directly from the road incident.

Overall then, none of the current solutions relies primarily and systematically on real-time information about the incident itself. It is thus questionable whether discrete data-driven approaches, such as the clustering method of Salamanis et al. (Salamanis et al., 2017), can solve the problem of abrupt changes to traffic correlation structure. Furthermore, as QTIP employs real-time simulations, it can be used as a component within other systems for real-time Dynamic Traffic Assignment, such as DynaMIT (Lu et al., 2015).

2.3.3 Model Selection

As mentioned above, the traffic models in QTIP can be freely chosen, hence this work focuses on proof-of-concept of the working principles of QTIP. We also note that, as observed in (Tune et al., 2016), the range of traffic models is already too large to examine in detail here. Consequently, we refrain from making particular recommendations on specific model types for modeling ordinary and abnormal traffic conditions. However, for completeness of description, we now provide several effective guidelines for model selection; a broader discussion of model selection is available in (Claeskens and Hjort, 2008).

Model selection entails a choice between several *model classes*, such as:

- *Parametric* vs. *non-parametric* modeling (Smith et al., 2002), namely, particular functional form vs. flexible form.

- *Simple* vs. *complex* models (Hoogendoorn and Bovy, 2001), e.g., as measured by number of parameters: a sufficient number is needed to properly generalize to unseen data, while an improper number might lead to under-fitting or over-fitting.
- *Interpretable* vs. *opaque* models (Wang et al., 2020): the former may allow better explanation of the captured patterns, but restricts the range of applicable models; the latter allows less in-depth insights, but admits a wider selection of models.
- *Standalone* models vs. *ensemble* methods (Li et al., 2014): the latter may improve accuracy by combining the strengths of multiple models, but also requires further weight tuning and might introduce unnecessary complexity.

Accordingly, we later experiment with models from several different classes. Moreover, as the method of *estimating model parameters* is also a matter of choice, we experiment with both Maximum Likelihood Estimation (Section 2.4.5) and Bayesian Inference (Section 2.5.6).

Once several models are selected for comparison, they can be compared using measures of *goodness of fit* (D’Agostino, 1986), including:

- *Prediction quality* analysis, e.g., via Mean Error measures or Coefficient of Determination.
- *Statistical tests* of similarity between fit and expected distribution, e.g., via Analysis of Variance or hypothesis tests, such as the Kolmogorov-Smirnov Test and Pearson’s Chi-Squared Test.

For example, in this work, we compare models based on minimization of MSD, MAE and RMSE (Section 2.5.4) between model predictions and test observations.

In addition, there exist various *Information Criteria* (IC) for model selection, based on theoretically optimal tradeoffs between model simplicity and fit accuracy (Tune et al., 2016). Different IC employ different assumptions about the modeled data, and popular IC examples include the Akaike Information Criterion, Bayesian Information Criterion, and Minimum Description Length. In this work, measures of prediction quality suffice for choosing between the models, hence we do not employ additional IC.

In summation, for traffic modeling in QTIP too, we advise to follow the systematic guidelines of model selection as above. Furthermore, we next describe

a proof-of-concept case study for evaluating QTIP, where we employ various models of speed prediction in a motorway. The case study thus constitutes an example of using QTIP itself to select between models through simulations, particularly for accident-prone roads. This simulation-based manner of model selection can be augmented by incorporating additional historical data about other roads in the traffic network, including records of past accidents.

We note that while speed prediction alone does not suffice for incident treatment by traffic practitioners, the principles of simulation-based model adaptation in the case study extend to other traffic models – such as traffic flow, trip delays, public transport disruptions, and risk of secondary incidents – which together are highly useful for incident management. The speed prediction models that we study can also provide road users with real-time information about the occurrence and effect of incidents, which is useful for route planning and estimation of trip time.

2.4 Case Study for Evaluating QTIP

In this section, we present experiments to evaluate the performance and capabilities of a specific application of QTIP. As a case study, we use the Hillerød Motorway, a highly utilized and often congested highway in Denmark, and compare the performance of a model adapted by QTIP vs. a non-adapted model in incident conditions.

2.4.1 Overall Design of Experiments

To evaluate the QTIP framework, we first need to decide which type of models $M_{ordinary}$ and $M_{abnormal}$ are used (see Figure 2.1 on page 13). We chose to evaluate three types of models separately: Linear Regression (LR), Deep Neural Network (DNN), and Gaussian Processes (GP). Whereas LR is linear and parametric, DNN and GP are non-linear and non-parametric. For each prediction model type, we pre-train $M_{ordinary}$ on a set of incident-free simulations, which we describe in the next section. In this manner, we generate different demand and road conditions in the training set, and can test how the proposed model types differ in performance.

Finally, we compare between the simple, non-adaptive model $M_{ordinary}$ and QTIP on several incident scenario experiments (see Figure 2.2 on the next page). In the present work, all considered incident scenarios are represented as blocks on a given road link. Each scenario differs in: (1) the location of road blocks in

Figure 2.2: In our experiments of incident conditions, we run QTIP to obtain $M_{abnormal}$ for different incident scenarios independently.

the link; (2) the location of road blocks in lanes; (3) the number of road blocks; and (4) the demand in terms of traffic volume.

For each incident scenario, we generate multiple replications, independently and with stochastic perturbations. Each replication serves as a single, independent experiment. To reflect real-time operation, QTIP receives in each experiment the simulated speed measurements, averaged in 1 min intervals, in three links: the link where road blocks would appear, its uplink, and its downlink, as in Figure 2.3 on the facing page.

Before the road blocks manifest, QTIP predicts future mean speeds through the already pre-trained $M_{ordinary}$, i.e. without adaptation. When the road blocks appear, QTIP receives notification of their location, similarly to IVMS distress signals. QTIP then quickly creates model $M_{abnormal}$ using "on-the-fly" simulations, and uses it for subsequent predictions.

To clarify, note that each experiment involves two different types of simulations. One is the simulation that emulates the ground truth, which provides QTIP with input of actual traffic. The other simulations are those which QTIP executes internally "on-the-fly" to construct $M_{abnormal}$. QTIP uses such internal simulations to consider "what-if" values of unknown incident parameters, which the distress signal does not indicate. For example, the distress signal does not carry the current demand level, hence while the actual demand may be "medium", QTIP internally considers all of "low", "medium", and "high" as possible demand levels in predictions. To keep this separation clear, we shall refer to the two types of simulations as "ground-truth simulations" and "what-if simulations", respectively.

Figure 2.3: The Study Link S where we simulate incidents on the Hillerød Motorway, its uplink U , and downlink D . Traffic on these links flows uni-directionally north-to-south.

2.4.2 The Road Network

The Hillerød motorway is located in Sjælland region in the Greater Copenhagen Area (Figure 2.4 on the next page) and is known for recurrent congestion and significant impacts of incidents on traffic conditions. 604 accidents were recorded in the past 5 years (2012-17) in the total 35.9 km of its length. Furthermore, several towns and cities served by this motorway – Farum, Værløse and Gladsaxe – have planned to complete several developments by 2020 (Danish Ministry of Transportation, 2017), which will increase significantly the traffic on the corridor.

For this work, we focus on the stretch between Farum N and Værebrovej in the North/South direction, as in Figure 2.4 on the following page. The test network is in total 11 km long and consists of signalized ramps, five interchanges and a two-lane carriage. 36.5% of all observed accidents (i.e. 220 records) in the past 5 years were in this stretch, 5.3% of which resulted in injuries and/or fatalities. The recurrent congestion and disruptions already motivated the implementation of a 3 km Hard Shoulder Running stretch, active during the morning peak hours (Lund Andersen, 2016).

2.4.3 Data for Calibration

To calibrate the simulated environment, we used three types of data, all from the Danish Road Directorate²: flows, average speed, and travel times, by link and in time intervals of 15 min (see Table 2.1 on the next page for vehicle volumes

²Mastra and Hastrid databases, <http://www.vejdirektoratet.dk/>

Figure 2.4: The Hillerød Motorway: data collection points as white circles (left), and locations of entries and exits (right).

Location	Entrance	Exit
Farum N	3171	Network start point
Farum C	544	No exit point
Værløse	937	100
Bagsværd	852	544
Værebrovej	351	312

Table 2.1: Vehicle counts, Monday 2016-Jun-20 7-8 AM

aggregated by hour). The data originates from eight online data collection points along the motorway, shown as white circles in Figure 2.4.

For this paper demonstration, it sufficed to gather a limited data set of field measurements. As a calibration set, we used measurements from the 7-8 AM period of the weekdays of June 20 to 24, 2016. As a separate validation set, we used the preceding week, namely June 13 to 17, 2016. As demand input for the simulator, we constructed two Origin-Destination (OD) matrices: one for light vehicles, and one for heavy vehicles. We also conducted on-field data collection to measure some network attributes, such as speed limits and signal timings.

2.4.4 Construction and Calibration of Simulated Environment

The simulator we use in this study is PTV VISSIM³. VISSIM is widely applied in practice for modeling of transportation systems, reproduction of freeway driving conditions, analysis of traffic operations, and studying of incident conditions (Gomes et al., 2004, Katrakazas et al., 2018).

For constructing the simulated network, we used OpenStreetMap⁴ and Google Earth⁵ along with the above-mentioned field observations. For creating the signal systems which control the inflow of the network, we used VISVAP, VISSIM's add-on for traffic signal controls and traffic management systems⁶. As no control pre-set configuration was available, we performed manual tuning of fixed signaling cycles to match both the field observations and the recorded flow data.

For car-following behavior, we used the Wiedemann 99 model as in (Aghabayk et al., 2013). This is controlled in VISSIM through parameters $CC0 \dots CC9$, together with two look ahead/back distribution parameters (PTV Group, 2017). For lane change behavior, we considered seven parameters, which represent acceleration distribution during lane changing manoeuvres, how far in advance each driver can anticipate the next exit/weaving/lane drop, and how aggressively that driver changes lanes to reach there. (Gomes et al., 2004).

We calibrated the simulated environment offline through a manual iterative process, as per (Park and Won, 2006), while following the Danish Guidelines (Danish Road Directorate, 2010) for model evaluation. Because microscopic traffic simulators are characterized by a large number of parameters (usually to represent different driving behaviors), we carried out a sensitivity analysis to select a subset of parameters to calibrate, following the method in (Manjunatha et al., 2013), which VISSIM also uses for similar performance measures (delays, flows and speeds). Note that in our simulated road network, there is essentially only one route from each origin to destination, thus relaxing the impact of route choice parameters.

For each iteration in the calibration process, we ran enough replications until reaching a level of confidence for the average travel times, mean speeds, and vehicle flows at the data collection points (Hollander and Liu, 2008). As Measures of Effectiveness (MOEs) for quality of calibration, we used Rooted Mean Squared Normalized Error (RMSNE) for speeds and travel times, and Geoffrey's

³<http://vision-traffic.ptvgroup.com/en-us/products/ptv-vissim/>

⁴<https://www.openstreetmap.org/>

⁵<https://www.google.com/earth/>

⁶<http://www.traffic-inside.com/tag/visvap-en/>

Parameter	Unit	Calibration Range	Calibrated Value
Desired Speed Distribution	km/h	50 ... 110	80 – 110
Reduced Speed Areas	km/h	20 ... 30	20
Emergency Stop Distance	m	5 ... 50	Varies by Link Connector
Lane Change Distance for Weaving (LC)	m	200 ... 500	Varies by Link Connector
Maximum Deceleration for Breaking	m/s ²	3 ... 8	3.0
Waiting Time Before Diffusion	s	10 ... 60	60
Standstill Distance (CC0)	m	1.5, 2.0, 2.5	3
Headway Time (CC1)	s	0.9, 1.0, 1.1	1
Safety Distance Reduction Factor		0.1 ... 0.6	0.1

Table 2.2: Final calibrated parameters.

E. Havers' value (GEH) for flows (Hollander and Liu, 2008):

$$\text{RMSNE} = (e_1/o_1 + \dots + e_n/o_n)^{0.5} / n^{0.5} \quad (2.1)$$

$$\text{GEH} = (2e_1^2/p_1 + \dots + 2e_n^2/p_n)^{0.5} \quad (2.2)$$

where for all $i = 1..n$: $e_i = m_i - o_i$ and $p_i = m_i + o_i$, such that o_i is the value actually observed and m_i is the corresponding value in simulation. As described previously in Section 2.4.3, we used a separate set of data measurements for validation.

From the sensitivity analysis, we identified 9 parameters which had the most impact on the Measures of Effectiveness. We have calibrated these parameters as summarized in Table 2.2.

Per (Florida Department of Transportation, 2014), a traffic model is acceptably calibrated when the MOEs yield $\text{RMSNE} < 0.15$ and $\text{GEH} < 5$. Eventually, we have obtained the following measurements of calibration quality:

- $\text{RMSNE} = 0.12$ for speeds and for travel times.
- $\text{GEH} < 2$ for traffic counts.
- Less than 1 min difference between simulated and observed mean travel times for each stretch of road, as shown in Figure 2.5c on the facing page. In particular, the end-to-end mean travel time that the calibrated model attains is 8.23 min, which is close to 8.6 min actually measured on-field.
- Stretch-to-stretch mean speeds close to measured speeds, as reflected in Figure 2.5a on the next page and Figure 2.5b on the facing page.

Figure 2.5: Results of calibration: speeds (2.5a, 2.5b) and travel times (2.5c). In 2.5c, "calibration" and "validation" correspond to the data sets defined in Section 2.4.3.

Finally, it is worth noting that several advanced offline calibration approaches have been proposed in the literature, tackling the complexities of large scale calibration (Zhang et al., 2017) and the high dimensionality of both input and output performance measures (Ciuffo and Azevedo, 2014). Yet, for the purpose of the case study at stake, the method we used above provided satisfactory performance.

2.4.5 Experiments

Having calibrated the simulated environment, we are now ready to execute experiments. Recall that we independently experiment with three different instances of QTIP: LR, DNN, and GP.

2.4.5.1 Details Common to All Experiments

Each QTIP experiment begins with constructing $M_{ordinary}$. To this end, we first run 150 incident-free simulations – 50 for each demand level: low, medium, and high – and fit $M_{ordinary}$ on the output of these simulations. Next, we proceed to run incident scenarios. In each scenario, an obstacle appears at 7:10 AM on

Figure 2.6: In our experiments, road blocks appear at one of three locations on the Study Link, and occupy one or two lanes.

the study link S , and disappears at 7:40 AM. This affects traffic not only in S , but also in its uplink U and downlink D , as shown in Figure 2.3 on page 19.

Figure 2.6 illustrates the various options for incident positions in our experiments, as following. Each incident scenario involves either 1 or 2 road blocks in one of three *locations* on S : Start, Center, or End. Each road block is positioned on one of the three *lanes* of S : Left, Middle, or Right. Two simultaneous road blocks occupy either the same lane, one right after the other (spaced by 10m), or different lanes, in which case they are located next to each other. In addition, each scenario involves one of three demand levels: low, medium, or high. Hence in total, the number of distinct incident scenarios is

$$81 = 3 \times 3 \times (B(3, 1) + B(3, 1) + B(3, 2)) , \quad (2.3)$$

where $B(a, b) = a!/(b!(a-b)!)$ is the binomial coefficient, and the terms on the right-hand side correspond respectively to the number of options for: demand, location, single blocked lane, two blocks in same lane, and two blocks in different lanes.

For each of these 81 incident scenarios, we generate in VISSIM 5 ground-truth simulations, replicated with perturbations, as we soon explain. Finally, we test QTIP independently on each ground-truth simulation in two different modes: one where the distress signals carry high precision location, and one where location precision is low. QTIP knows which specific lanes are blocked only in high precision mode, in which case it could possibly yield more accurate predictions.

When QTIP receives notification of an incident, it generates 100 what-if simulations, and uses their results to fit $M_{abnormal}$ as a piece-wise prediction model.

The first piece pertains to the first 6 critical minutes immediately after the appearance of road blocks, and the second piece pertains to the time until the incident is cleared. $M_{abnormal}$ also takes into account the number of minutes which have passed since incident occurrence, as variable $T_{accident}$.

To recap so far, three sets of simulations are involved: incident-free (to train $M_{ordinary}$), incident ground-truth (used as benchmark "real" measurements), and incident what-if (created by QTIP to train $M_{abnormal}$). To account for the stochastic nature of traffic, we apply perturbations to each simulation as follows: before running any simulation, we perturb its input OD matrix A independently as $\tilde{A}$, so that $\tilde{A}_{ij} = \tilde{c}_{ij} A_{ij}$, where $\tilde{c}_{ij} \sim \mathcal{N}(1, 0.2)$. To simulate different demand levels, we also scale $\tilde{A} = \hat{c}\tilde{A}$, so that $\hat{c} = 0.7, 1.0, 1.3$ for low, medium, and high demand, respectively. The simulation then runs with $\tilde{A}$ as its OD matrix. Note that medium demand corresponds to the same demand as we have measured on-field earlier, with an added stochastic component.

All simulations yield as output 1 min mean speeds in links S, D, U (Figure 2.3 on page 19). Models $M_{ordinary}$ and $M_{abnormal}$ receive this output in the form of vectors, as following. At any time point, let S_k, U_k, D_k denote the 1 min mean speed k minutes earlier in links S, U, D , respectively. For each of 6:50, 6:51, $\dots$, the corresponding vector is $[S_5, S_6, U_5, U_6, D_5, D_6]$, and S_0 is the response variable. That is, to predict the 1 min mean speed on S , the models use the 1 min mean speeds 5 and 6 min beforehand on links S, D, U . The vectors do not include speeds earlier than 6 min, because adding such information did not improve prediction quality in our experiments.

2.4.5.2 Experiments with Linear Regression

Linear Regression (LR) assumes the following linear relationship between the response variable S_0 and the explanatory variables:

$$S_0 = \beta_5^S S_5 + \beta_6^S S_6 + \beta_5^U U_5 + \beta_6^U U_6 + \beta_5^D D_5 + \beta_6^D D_6, \quad (2.4)$$

where β 's are trainable parameters. We train this model using Ordinary Least Squares (OLS) to obtain the parameters that minimize the sum of squared differences between observed and predicted values of S_0 , as detailed in (Weisberg, 2005).

2.4.5.3 Experiments with Gaussian Processes

Whereas LR is a parametric modeling method, Gaussian Process (GP) regression is non-linear and non-parametric. GP assumes that the relationship be-

Figure 2.7: Structure of Deep Neural Networks in our experiments.

tween S_0 and the explanatory variables is an unknown function, drawn from a multivariate Gaussian distribution,

$$\mathcal{N}(\mathbf{0}, \mathbf{K}), \quad (2.5)$$

where for the given n observations, $\mathbf{K}$ is a n -by- n covariance matrix, so that for any two vectors x_i, x_j ($i, j \in 1..n$), $\mathbf{K}_{i,j}$ expresses the similarity between x_i and x_j .

As a prior distribution, we define the elements of $\mathbf{K}$ through the commonly used RBF kernel function,

$$\mathbf{K}_{i,j} = \exp(-0.5 l_2(x_i/\tau, x_j/\tau)^2), \quad (2.6)$$

where $l_2(\cdot)$ denotes the Euclidean distance norm, and τ is a scaling factor. To cover a wide range of scaling factors, we experiment with $\tau \in \{0.1, 1, 2, 4, 8, 9, \dots, 16\}$. Each model is then trained through Maximum Likelihood Estimation (MLE), as detailed in (Williams and Rasmussen, 2006).

2.4.5.4 Experiments with Deep Neural Networks

The Deep Neural Network (DNN) models in our experiments are parametric and non-linear, and are structured as in Figure 2.7. These models pass an input vector $x = [S_5, S_6, U_5, U_6, D_5, D_6]$ through a succession of densely connected hidden layers, and finally output the sum of the results as the predicted value of S_0 . Each of the h hidden layers consists of g sigmoid units, each structured as:

$$\sigma(x) = \frac{1}{1 + \exp(-w^T x)}, \quad (2.7)$$

where w is a vector of trainable parameters, and h, g are hyper-parameters that we specify below. For example, if $h = 2$ and $g = 5$, then the DNN models the

relationship between x and S_0 as:

$$\begin{aligned} S_0 = & \sigma_{1,2}([\sigma_{1,1}(x), \dots, \sigma_{5,1}(x)]) + \\ & \sigma_{2,2}([\sigma_{1,1}(x), \dots, \sigma_{5,1}(x)]) + \\ & \dots + \\ & \sigma_{5,2}([\sigma_{1,1}(x), \dots, \sigma_{5,1}(x)]) , \end{aligned} \quad (2.8)$$

where $\sigma_{i,j}$ denotes the i 'th sigmoid in the j 'th hidden layer.

We have experimented with $h \in \{1, 2\}$ hidden layers, each consisting of $g \in \{5, 10\}$ sigmoids. Each model is trained with backpropagation for a maximum of 100 epochs, using Mean Squared Error as loss function, mini batches of size 100, the Adam optimizer, and a 10% validation split to monitor overfitting. For more details of this training procedure, we kindly refer the reader to (Kingma and Ba, 2014).

2.4.5.5 Model Complexity

Before proceeding to the experimental results, let us compare the three model types in terms of computational complexity. Generally speaking, LR is the simplest of the three because it incorporates few parameters. DNN and GP, however, are not directly comparable: GP is non-parametric and scales cubically with the number of input vectors, whereas the complexity of fitting DNN models depends on the number of epochs and mini-batches. The complexity of prediction, however, is quite similar across all model types, i.e., it takes roughly the same time to generate predictions from any of the models once trained.

2.5 Results

We now provide the results of the experiments that we constructed in Section 2.4. The results will show that for a wide range of incident cases, the information in IVMS signals suffices for $M_{abnormal}$ to reduce prediction errors significantly and timely. While analyzing the results, we also provide some insights into the behavior of traffic under incidents, and examine how prediction quality deteriorates if model adaptation is not performed.

2.5.1 Real-Time Performance of QTIP

We executed all experiments on a server with 16 GB of memory and an Intel i7-2600 CPU, clocked at 3.40 GHz. It took QTIP at most 45s to simulate

Model Type	Mean RMSE	
	Lanes Known	Lanes Unknown
GP $\tau = 8$	13.928	23.676
DNN $h = 2, g = 10$	6.328	9.145
LR	6.290	8.878

Table 2.3: Best performing model of each type. Values closer to zero are better, best values are highlighted in bold.

15 min, for any single what-if simulation. Once QTIP has executed all what-if simulations, it required only a few more seconds to fit an LR model $M_{abnormal}$ on their results. Hence when all what-if simulations are executed in parallel, QTIP can yield an adapted model $M_{abnormal}$ within 1 min, just in time for handling the first critical minutes of the incident. It should also be noted that recently developed methods for GPU acceleration of micro-simulations (Heywood et al., 2018) may further cut down these running times.

2.5.2 Comparison of Predictive Performance

For each model type among GP, DNN, and LR, Table 2.3 summarizes the predictive quality of the best performing model as measured through mean RMSE (2.11) over all experiments. As may be expected, all models perform better when blocked lanes are known than otherwise.

LR is the best performing model type when lanes are either known or unknown. GP performs much worse than both LR and DNN, possibly because of inappropriate choice of kernel function or overfitting to train data. The poor fit of GP may thus be alleviated through, e.g., incorporating a periodic kernel function or applying regularization techniques (Roberts et al., 2013). The predictive performance of DNN is a little worse than that of LR, as we next explain.

The small differences in predictive performance between LR and DNN could be caused by a number of reasons. First, the dataset at hand may conform better to linear (LR) than non-linear (DNN) patterns, although this could change if more explanatory variables are added or if substituting for a larger dataset. Second, while the training methods of both models are based on stochastic optimization, training a DNN through backpropagation typically requires more delicate tuning than training LR through OLS. It could thus be that finer selection of DNN hyper-parameters, such as via Bayesian Optimization (Snoek et al., 2012), would yield a better performing DNN model. Finally, the DNN models can be made more powerful through a change of structure, e.g., by stacking Recurrent units for time series data (Ho et al., 2002), thus adding memory capabilities that LR

does not possess.

While further improvement of the models is intriguing in itself, it is nevertheless superfluous for the objectives of this paper. First, LR, DNN, and GP are commonly used model types for traffic modeling, and they demonstrate (Table 2.3 on the facing page) that higher precision of incident location indeed yields better predictions across all model types. Second and more importantly, the particular choice of model types is less important for proof-of-concept purposes, as the results we next present successfully convey the fundamental message of this paper: that model degradation under incidents can be mitigated through just-in-time, simulation-based adaptation. Third, as explained in Section 2.2.3, the main source of complexity for modeling under incidents is, in any case, the sudden change of variable correlation structure, regardless of model form. We thus defer model improvements to future work (Section 2.6.3), and focus next on the predictive performance of QTIP with LR, the best performing model.

2.5.3 Closer Look at Several Representative Incident Scenarios

While analyzing the results of our experiments, we have observed several different behaviors of the mean speed under incidents. We now illustrate these behaviors by selecting and visualizing several representative scenarios, for both cases of known and unknown blocked lanes. For each representative scenario, we plot in Figure 2.8 on the next page the time series of 1 min ground-truth speed, averaged over all ground-truth simulations of that scenario. For further analysis, we also plot the predictions of $M_{abnormal}$ when applied to each of these averaged time series.

First, we see in Figure 2.8 on the following page that mean speed in the Study Link S mostly drops after the onset of an incident (2.8b, 2.8d, 2.8e, 2.8f), but less frequently, it can also gradually increase (2.8a). This exceptional phenomenon occurs when the road blocks appear at the start of the link, so that vehicles which eventually enter the link can then flow freely. We also see that the standard deviation of mean speed typically exhibits more variation when only 1 lane is blocked (2.8a, 2.8b, 2.8d, 2.8e) than when 2 different lanes are blocked (2.8f). This is because 2 blocked lanes bring about a level of congestion, such that queued vehicles travel in similarly low speeds, whereas speeds are more variable when only 1 lane is blocked. Finally, we see that the predictions of $M_{abnormal}$ often come close to the actual mean speed during the incidents. Indeed, we next show that $M_{abnormal}$ mostly outperforms $M_{ordinary}$ in incident scenarios.

Figure 2.8: Representative examples of behavior of mean speed in Study link S under different incident scenarios, and performance of $M_{abnormal}$.

2.5.4 Model for Ordinary Conditions Degrades under Incidents

Let us now measure what happens if QTIP is unavailable, so that only $M_{ordinary}$ is used for traffic prediction. To this end, we compare the performance of LR $M_{ordinary}$ on incident-free simulations vs. incident simulations. The performance measurements we use are: Mean Signed Deviation (MSD), Mean Absolute Error (MAE), and Rooted Mean Squared Error (RMSE). These are defined as following for any set of vectors $V = \{v_1, \dots, v_N\}$, where for $v_i \in V$, r_i is the model estimation minus the actual value of the response value.

$$\text{MSD} = (r_1 + \dots + r_N) / N, \quad (2.9)$$

$$\text{MAE} = (|r_1| + \dots + |r_N|) / N, \quad (2.10)$$

$$\text{RMSE} = ((r_1^2 + \dots + r_N^2) / N)^{0.5}. \quad (2.11)$$

Figure 2.9: Performance of $M_{ordinary}$ under normal and abnormal traffic conditions in our experiments. All plots show predicted vs. ground-truth mean speed, such that accurate predictions lie on the diagonal line. Under normal conditions, $M_{ordinary}$ performs better when its input contains information about the uplink U and downlink D (2.9a) than without this information (2.9b). For incident conditions, 2.9c shows the density of scattered points. The predictive performance of $M_{ordinary}$ degrades considerably under incidents, and it tends to over-estimate the mean speed, as also reflected in the Mean Signed Deviation (MSD).

Conditions	U and D	MSD	MAE	RMSE
Normal	with	-0.183	3.555	4.839
	w/o	-0.025	3.693	5.259
Incident	with	3.040	6.862	9.513
	w/o	0.440	5.913	9.085

Table 2.4: Performance of $M_{ordinary}$ under normal vs. incident conditions

For incident-free simulations, we apply 10-fold cross validation, and obtain Figure 2.9, which illustrates the prediction accuracy of $M_{ordinary}$ for each vector from incident-free simulations. We see that $M_{ordinary}$ is mostly accurate in predicting speeds under incident-free conditions, as we might expect, because traffic in highways flows in a rather regular manner when no disruptions occur.

Next, we measure how worse $M_{ordinary}$ performs on incident conditions, for which it was not trained. The results are summarized in Table 2.4, and we see that $M_{ordinary}$ deteriorates in comparison with its performance under normal conditions. The deterioration occurs whether or not $M_{ordinary}$ has access to information about uplink U and downlink D . For the former option, Figure 2.9c depicts the tendency of $M_{ordinary}$ to overestimate the actual mean speed in study link S .

To further reason about this deterioration, we next visualize how $M_{ordinary}$

Figure 2.10: Degradation of $M_{ordinary}$ when applied to the average incident time series, either with or without input from uplink U and downlink D .

performs on an the average incident time series, where each lag is the 1 min mean speed, averaged over all ground-truth simulations. Figure 2.10 shows how $M_{ordinary}$ fails to predict the average incident time series, either with or without information about U and D . In the former case, the predictions of $M_{ordinary}$ on the average incident time series do not even converge to the typical speed at the second phase of the incidents. Upon examining simulations output, it seems that this lack of convergence is caused by an increase in mean speed in D while congestion forms in S .

In fact, we also see in Figure 2.10 the three typical phases of the mean speed under road incidents, as following. The first phase immediately follows the onset of the incident, at which time the mean speed drops sharply for a few minutes. In the second phase, the mean speed stabilizes, while the incident is still in place. The third phase immediately follows the clearing of the incident, at which time the mean speed increases sharply for a few minutes, resuming the trend it had just before the incident.

2.5.5 Improvement of Predictive Performance with QTIP

We have shown that $M_{ordinary}$ deteriorates significantly under incident conditions, which further expresses the need for just-in-time model adaptation. Therefore, we now measure the performance gain when using the adapted model $M_{abnormal}$ instead of $M_{ordinary}$ in the first 6 min of each incident scenario. To this end, we calculate the relative RMSE improvement for each scenario, as:

$$\frac{\text{RMSE}(M_{ordinary}) - \text{RMSE}(M_{abnormal})}{\text{RMSE}(M_{ordinary})}, \quad (2.12)$$

where $\text{RMSE}(M)$ is the RMSE of predictions of model M for the corresponding scenario. Positive values correspond to lower prediction error of $M_{abnormal}$ vs. $M_{ordinary}$, namely better performance of $M_{abnormal}$.

Figure 2.11: Relative improvement of RMSE of $M_{abnormal}$ over $M_{ordinary}$ in the first 6 min of incidents, when QTIP either knows or does not know which lanes are blocked. All models use Linear Regression, and higher values correspond to greater improvement.

Figure 2.11 visualizes the relative RMSE improvement for each incident scenario. We see that $M_{abnormal}$ mostly outperforms $M_{ordinary}$, whether or not QTIP knows the exactly blocked lanes. Moreover, we see that when QTIP knows which lanes are blocked, the prediction quality of $M_{abnormal}$ considerably increases. Averaging all the values in Figure 2.11, we obtain that the mean relative RMSE improvement over all incident scenarios is 28.74%.

We also see in Figure 2.11 a few exceptional cases, in which $M_{ordinary}$ outperforms $M_{abnormal}$. Most of these cases have the following in common: two adjacent vehicles block the same lane, demand is low, and location precision is low. Such circumstances are illustrated in Figure 2.8c on page 30, where we see that traffic disruption is then rather minor, so that the behavior of the mean speed remains rather stable when the incident occurs. Hence in such circumstances, on one hand, the uninformed $M_{ordinary}$ performs well. On the other hand, QTIP does not know that the two distress signals originate from vehicles on the same lane, and so generates what-if simulations also for two different blocked lanes. Consequently, in such circumstances, QTIP trains $M_{abnormal}$ to predict a disruption greater than actual.

2.5.6 Transfer Learning

In the above experiments, $M_{abnormal}$ is fit afresh for each scenario. Nevertheless, as $M_{ordinary}$ incorporates knowledge about historical traffic, it could be beneficial to give $M_{abnormal}$ access to this knowledge during training. In other words, $M_{abnormal}$ could possibly benefit from some form of transfer learning (Pan and Yang, 2009).

To explore this possibility, we next carry out additional experiments, where the linear coefficients of $M_{abnormal}$ are based on the coefficients of $M_{ordinary}$. In these additional experiments, we again use the same 81 incident scenarios as in Section 2.4.5.1, when QTIP either knows or does not know the precisely blocked lanes. This time, however, we use *Bayesian Inference* (Peled et al., 2019c) to obtain $\beta \in \mathbb{R}^k$, the linear coefficients of $M_{abnormal}$.

2.5.6.1 Bayesian Setup

In each experiment, the *prior* on the coefficients of $M_{abnormal}$ is a multivariate Gaussian,

$$p(\beta) = \mathcal{N}(\beta \mid \mu, \sigma_\beta^2 \mathbf{I}_k) , \quad (2.13)$$

where $\mu \in \mathbb{R}^k$ is the coefficients of $M_{ordinary}$, $\mathbf{I}_k$ is the $k \times k$ identity matrix, and σ_β is a hyper-parameter. Hence without further evidence about incidents, $M_{abnormal}$ is initially similar to $M_{ordinary}$. When such evidence is given as features $\mathbf{X} \in \mathbb{R}^{n \times k}$ and corresponding observations $\mathbf{y} \in \mathbb{R}^n$, the *likelihood* of the observations is

$$p(\mathbf{y} \mid \mathbf{X}, \beta, \sigma_y) = \mathcal{N}(\mathbf{y} \mid \mathbf{X}\beta, \sigma_y^2 \mathbf{I}_n) , \quad (2.14)$$

where σ_y is another hyper-parameter. In this Section, we fix $\sigma_\beta = 1$ and $\sigma_y = 1$; we have also experimented with significantly higher and lower values of σ_β and σ_y , but obtained no noticeable change in results.

By (2.113)–(2.117) in Section 2.3 of (Bishop, 2006), the *posterior* on $M_{abnormal}$ coefficients is

$$p(\beta \mid \mathbf{y}) = \mathcal{N}\left(\beta \mid \left\{ \mathbf{X}^T (\sigma_y^{-2} \mathbf{I}_n) \mathbf{y} + (\sigma_\beta^{-2} \mathbf{I}_k) \mu \right\}, \right) , \quad (2.15)$$

where $\in \mathbb{R}^{k \times k}$ is

$$= \left(\sigma_\beta^{-2} \mathbf{I}_k + \mathbf{X}^T (\sigma_y^{-2} \mathbf{I}_n) \mathbf{X} \right)^{-1} . \quad (2.16)$$

We thus use the posterior mean of (2.15) as the fitted coefficients of $M_{abnormal}$.

2.5.6.2 Experiments and Results

The experiments in this Section allow us to study how an increasing amount of simulated incident information affects Bayesian $M_{abnormal}$ vs. freshly fit $M_{abnormal}$. For each incident scenario i , we first independently generate 5 ground truth simulations of the incident, which we later use for evaluating the models. Then for each i and $j = 1, 2, \dots, 10$, we have QTIP independently generate j what-if simulations, on which we train each model. Because each training simulation involves stochastic variations, we repeat the experiment 30 times for each i, j , evaluate the RMSE of each model's predictions vs. the ground truth observations, and average the 30 evaluations.

Figure 2.12 on the following page depicts the results for known lanes, whereas Figure 2.13 on page 37 depicts the results for unknown lanes. Each plot corresponds to a different incident scenario and illustrates RMSE for Bayesian $M_{abnormal}$, fresh $M_{abnormal}$, and $M_{ordinary}$, which does not utilize incident simulations and thus appears fixed. We limit the plots to $j = 1..5$ training simulations, because both $M_{abnormal}$ models typically perform closely for $j \geq 6$.

We see that compared to fresh $M_{abnormal}$, Bayesian $M_{abnormal}$ mostly obtains lower average RMSE and lower standard deviation for $j \leq 4$ simulations. Hence by transferring knowledge from $M_{ordinary}$, Bayesian $M_{abnormal}$ can take better advantage of a low number of training simulations than can fresh $M_{abnormal}$, which has no such knowledge.

However, there are a few occasions where modeling afresh is more advantageous, e.g., scenarios “L,S,B,B” in Figure 2.12 on the following page and “L,S,T,T” in Figure 2.13 on page 37. In addition, when QTIP generates exceedingly few training simulations ($j \leq 2$), $M_{ordinary}$ often outperforms both $M_{abnormal}$ models. In conclusion, transfer learning can be advantageous in situations where QTIP resources are reasonably limited, e.g., when only a few computational nodes are available for parallel execution of what-if simulations.

2.6 Summary of Key Points and Future Work

We have presented the QTIP framework for real-time model adaptation under nonrecurrent traffic disruptions. QTIP is motivated by both a problem and an opportunity. The problem is that traffic prediction models must be adapted in real-time to properly deal with abnormal road conditions, yet current solutions fall short of addressing this need. The opportunity arises from In-Vehicle Monitoring Systems (IVMS), which provide immediate indication and information about incident occurrence.

Figure 2.12: Average RMSE (km/h, vertical axis) as the number of training simulations increases (horizontal axis), when QTIP knows which lanes are blocked. The shaded areas are ± 1 standard deviation around the average. Titles pertain to incident scenarios, as: demand level (High, Medium, Low), location on link (Start, Center, End), 1st and 2nd blocked lanes (Top, Middle, Bottom, None).

Figure 2.13: Average RMSE (km/h, vertical axis) as the number of training simulations increases (horizontal axis), when QTIP does not know which lanes are blocked. The shaded areas are ± 1 standard deviation around the average. Titles pertain to incident scenarios, as: demand level (High, Medium, Low), location on link (Start, Center, End), 1st and 2nd blocked lanes (Top, Middle, Bottom, None).

Under incident conditions, QTIP generates the data required for model adaptation from real-time simulations of the affected road. The simulations take advantage of IVMS as source of real-time incident information, while QTIP allows free choice of the complementary data-driven prediction models. Our solution methodology thus combines two traditionally distinct approaches to problem modeling: “black box” machine learning algorithms on one hand, and “white box” transport engineering methods on the other hand.

2.6.1 Findings and Implications

To evaluate QTIP, we have devised a proof-of-concept case study where incident conditions are represented as sudden road blocks on a major motorway in Denmark. We have then experimented QTIP with several model types: Linear Regression, Gaussian Processes, and Deep Neural Networks. Following are our main empirical findings.

1. Our results verify measurably both the degradation in predictive quality if no model adaptation is performed, and the gain in predictive quality when QTIP is used for model adaptation (Section 2.5.4). Additionally and as may be expected, the adapted prediction model improves when more incident parameters are known, regardless of model type (Section 2.5.5).
2. In most cases, the adapted model outperforms the non-adapted model, so that the mean relative RMSE improvement over all cases is 28.74% (Figure 2.11 on page 33). There are also a few edge cases, where the non-adapted model performs better than QTIP, as we explain in Section 2.5.5.
3. As each simulation runs in under 1 min on our single-PC platform, QTIP could yield an adapted model in real-time through parallel execution (Section 2.5.1).

Our findings thus imply that the long-standing problem of instantaneous model adaptation under incident occurrence is becoming more tractable, as In-Vehicle Monitoring Systems are increasingly deployed. This improvement in accuracy further implies real-world benefits for more effective incident management, and in particular, our findings suggest considerable positive impact in circumstances of high traffic demand, during which efficient traffic management is most important. Nevertheless, it is hard to assess the impact of this paper on traffic policy making, e.g., in terms of passenger time savings or the gains of quicker emergency response. Rather, this is a proof-of-concept study with the objective of delivering a key theoretical message: that the suggested solution approach promotes the prospect of just-in-time prediction model adaptation.

2.6.2 Limitations

Every prediction framework has its limitations, and so does QTIP. For demonstrating the full potential of QTIP, this work examines the best case, namely 100% availability of IVMS signals. Indeed, global trends suggest high adoption rate of IVMS in upcoming years. In addition, QTIP currently depends on an external module for calibration of simulations. Nevertheless, roads which are particularly prone to incidents may be pre-calibrated offline. It should also be noted that IVMS signals alone cannot resolve an inherent difficulty of traffic prediction: that full observability of network state requires a number of road sensors much larger than is typically available. As such, any solution based wholly on simulations may yield a sub-optimal adapted model, no matter how many simulations it employs.

2.6.3 Future Work

The above discussion raises several interesting directions for future work:

- It is expected that the adoption of IVMS technology will happen gradually. Thus, the performance of QTIP may be measured under varying, lower rates of availability of such signals.
- The current QTIP framework can be extended to other test cases, i.e. road types, road conditions, and larger networks, to test e.g. other demand conditions and effects of route choice. It can also be extended to other traffic disruption types. All of these extensions may ultimately yield different predictive performances in QTIP.
- In large-scale network scenarios, the computational performance of QTIP may worsen. Methods for parallel simulations, sub-network selection, or multi-scale frameworks can be tested within QTIP to achieve timely prediction improvements.
- The integration of QTIP with existing systems for real-time traffic management (Kong et al., 2013, Lu et al., 2015) can be piloted, to assess performance in practical deployment conditions.
- The data-driven prediction models used in this paper can be further enhanced as described in Section 2.5.2, and additional model types can be experimented with, e.g., ARIMA or Gradient Boosting (Zhang and Haghani, 2015).
- Finally, we also point out that the integration of online calibration within QTIP may provide increased prediction power to the overall framework

(Figure 2.1 on page 13). The past 15 years have been rich in online calibration of meso- and microscopic simulators (Antoniou et al., 2007, Prakash et al., 2018, Qin and Mahmassani, 2004), and further work is needed to evaluate real-time performance and additional benefits of QTIP with online calibration.

Acknowledgment

The research leading to these results has received funding from the People Programme (Marie Curie Actions) of the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Individual Fellowship H2020-MSCA-IF-2016, ID number 745673.

Modeling Latent Mobility Demand via Censored Gaussian Processes

This Chapter is based on a joint work with several DTU researchers: PhD student, Daniele Gammelli; his supervisors, Assoc. Profs. Filipe Rodrigues and Dario Pacino; and my supervisor, Prof. Francisco C. Pereira. The work was published in November 2020 in *Transportation Research Part C: Emerging Technologies* (Gammelli et al., 2020), and pertains to censorship of shared mobility data, wherein available supply inherently limits observed demand. We propose to account for this censorship non-parametrically via Gaussian Processes, for which we derive a censored likelihood function, and we show empirically that this method outperforms common approaches for handling censorship.

Daniele is the main author: he came up with the methodology and conducted experiments with both synthetic datasets and a real-world bike-sharing dataset. As a second author, I extended the methodology by formulating a supply-based scheme for stochastic censorship of short-term demand, which I applied on real-world taxi pickups and dropoffs. I also wrote the Literature Review on censored modeling and parts of the Introduction and Summary. To rebut a reviewer comment, I further conducted experiments with matrix-based imputation, as attached in Section 3.6.1.

Within the theme of my thesis, this Chapter studies Gaussian Processes as a flexible Machine Learning method for dealing with uncertainty in partially observed mobility demand, where the differences between censored observations and true values are unknown. For me, this work underlined that data censorship is prevalent and requires explicit treatment in Transportation modeling. Consequently, in the second half of my PhD studies, I directed more research effort into censored modeling, which is also the subject of the continuation work in Chapter 4.

3.1 Introduction

Being able to understand and predict demand is essential in the planning and decision-making processes of any given transport service, allowing service providers to make decisions coherently with user behavior and needs. Having reliable models of demand is especially relevant in shared transport modes, such

as car-sharing and bike-sharing, where the high volatility of demand and the flexibility of supply modalities (e.g., infinitely many possible collocations of a car-sharing fleet) require that decisions be made in strong accordance with user needs. If, for instance, we consider the bike sharing scenario, service providers face a great variety of complex decisions on how to satisfy user demand. To name a few, concrete choices must be made for what concerns capacity positioning (i.e., where to deploy the service), capacity planning (i.e., dimensioning the fleet size), rebalancing (i.e., where and when to reallocate idle supply), and expansion planning (i.e., if and how to expand the reach of the service).

Demand modeling uses statistical methods to capture user demand behavior based on recorded historical data. However, historical transport service data is usually highly dependent on *historical supply* offered by the provider itself. In particular, supply represents an upper limit on our ability to observe realizations of the true demand. For example, if we have data about a bike-sharing service with 100 bikes available, we might observe a usage (i.e., demand) of 100 bikes even though the actual demand might have potentially been higher. This leads to a situation in which historical data is in fact representing a biased, or *censored*, version of the underlying demand pattern in which we are truly interested. More importantly, using censored data to build demand models will, as a natural consequence, produce a biased estimate of demand and an inaccurate understanding of user needs, which will ultimately result in non-optimal operational decisions for the service provider.

To address these problems, we propose a general approach for building models that are aware of the supply-censoring issue, and which are ultimately more reliable in reflecting user behavior. Inspired by current censored modeling literature, we formulate a *censored* likelihood function and apply it within a Gaussian Process model of transport demand. Using both synthetic and real-world datasets of shared mobility services, we pit this model against non-censored models and analyze the conditions under which it is better capable of recovering true demand.

3.2 Review of Related Work

3.2.1 Demand Forecasting for Shared Mobility

There exists a long line of literature about demand forecasting, with research mainly focusing on obtaining predictions of demand based on historical data in combination with some exogenous input, such as weather and temporal information. The bike-sharing context alone already provides many such examples,

as follows. Du et al. (2019), Yang et al. (2016b) use Random Forest models for rental predictions, whereas Rudloff and Lackner (2014) approach the demand forecasting problem through the use of Generalized Linear Models (GLMs) of counts, namely, Poisson and Negative-Binomial regression models. Rixey (2013) apply multivariate regression using data gathered from multiple bike-sharing systems.

Li et al. (2015) define a hierarchical prediction model, based on Gradient Boosting Regression Trees, for both rentals and returns at a city-aggregation-level. They model rent proportions for clusters of bike-stations via multi-similarity-based inference and predict returns consistently with rentals by learning transition probabilities (i.e., where bikes end after being rented). A different approach is found in (Singhvi et al., 2015), where the proposed prediction model is based on log-log regression and refers to the Origin-Destination (O-D) trip demand, rather than rentals and returns in specific locations or areas. Lin et al. (2018), Xu et al. (2018), Zhang et al. (2016) instead propose deep learning-based approaches to model bike in-flow and out-flow for the bike-sharing system areas and station-level demand respectively.

3.2.2 Censored Modeling

Data censorship involves two corresponding sets of values: true and observed, so that each observed value may be *clipped* above or below the respective true value. Observations that have not been clipped are called *non-censored*, and correspond exactly with true values. All other observations are called *censored* as they have been clipped at their observed values, therefore giving only a limited account of the corresponding true values (e.g., observed demand not corresponding with true demand). We next review methods for modeling censored data, namely, *censored modeling*.

An early form of censored modeling is the Probit model (Aldrich et al., 1984) for binary (0/1) observations, which assumes that the probability of observing 1 depends linearly on the given explanatory features. James Tobin extended this to a model for censored regression (Tobin, 1958), now called Tobit, which assumes that the latent variable depends on the explanatory features linearly with Gaussian noise, and that observations are censored according to a fixed and known threshold. Because the censored observations divert Least Squares estimators away from the true, latent values, Tobit linear coefficients are instead commonly estimated via Maximum Likelihood techniques, such as Expectation Maximization or Heckman’s Two-Step estimator (Amemiya, 1984).

The Tobit model has since become a cornerstone of censored modeling in multi-

ple research fields, including econometrics and survival analysis (Bewick et al., 2004). It has been extended through multiple variations (Greene, 2011), such as: multiple latent variables, e.g., one variable determines which observations are non-censored while another determines their values (Amemiya, 1985); Tobit-like models of censored count data (Terza, 1985); Tobit Quantile Regression (Powell, 1986); dynamic, autoregressive Tobit (Wei, 1999); and combination with Kalman Filter (Allik et al., 2015). Other methods of censored regression have also been suggested, predominantly for survival analysis, such as: Proportional Hazard models (Kay, 1977), Accelerated Failure Time models (Wei, 1992); Regularized Linear Regression (Li et al., 2016); and Deep Neural Networks (Biganzoli et al., 2002, Wu et al., 2018).

As reflected in all the above works, research into censored modeling commonly aims to reconstruct the latent process that yields the true values. In this work too, we aim to predict true values, not the occurrence of clipping nor the actually observed values. Similarly to Tobit, we too assume that all observations are independent and that each observation is known to be either censored or non-censored.

A method related to our work is found in (Groot and Lucas, 2012), where Gaussian Processes are used to model censored artificial data characterized by a fixed and known threshold. Our work builds upon this method and extends it to time-varying thresholds, because in real-world scenarios, the observability of demand is typically limited by time-varying supply.

A theoretical treatment of non-parametric censored regression appears in (Lewbel and Linton, 2002), where the authors derive estimators for the latent variable with fixed censoring, analyze them mathematically, and apply them to a single, simulated dataset. In contrast, this work proposes a likelihood function and applies it to multiple, real-world datasets with dynamic censoring.

3.2.3 Common Approaches to Handling Demand Censorship

Within the above stream of research, the censoring problem discussed in Section 3.1 is widely accepted. However, to the best of our knowledge, there has been no extensive study on how observed demand can differ from the true, underlying demand and how these differences could impact predictive models. To assess this issue, common approaches regard various *data cleaning techniques*. For example, Albiński et al. (2018), Goh et al. (2019), O’Mahony and Shmoys (2015), Rudloff and Lackner (2014) attempt to avoid the bias induced by censored observations by filtering out the time periods where censoring might have occurred, before modeling. As a further example, Albiński et al. (2018) substitute the censored

observations with the mean of the historical (non-censored) observations regarding the same period. A different but related approach is proposed by Freund et al. (2019), Jian et al. (2016) who focus on obtaining an unbiased estimate of arrival rate by omitting historical instances where bikes were unavailable.

These common approaches manage, to some degree, to correct the bias characterizing observed demand. However, they also give relevance to the fact that this problem represents an important gap which requires further study in order to obtain reliable predictions for shared transport and other fields. We believe there are two main reasons to why there is the need for a more structured view on the censoring problem: (i) the approaches described above might not be applicable in many real-world scenarios in which the number of censored observations is very high, leading either to an excessive elimination of useful data or to an inadequate imputation of the censored data; (ii) rather than resorting to cleaning and imputation procedures as the ones above, it would be desirable to equip the forecasting model with some sort of awareness towards the censoring problem, so that the model can utilize the entire information captured in the observations to coherently adjust its predictions.

3.3 Methodology

In this Section, we incrementally describe the building blocks of our proposed censored models. First, we introduce several general concepts: Tobit likelihood, Gaussian Processes, and the role of *kernels*. Then, we combine these concepts by defining Censored Gaussian Processes along with a corresponding inference procedure.

3.3.1 Tobit Likelihood

As a reference point for developing our censored likelihood function, let us now elaborate on the likelihood function of the popular Tobit censored regression model, described in Section 3.2.2. For each observation y_i , let y_i^* be the corresponding true value. For instance, in a shared transport demand setting, y_i^* is the true, latent demand for shared mobility, while y_i is the observed demand; if y_i is non-censored then $y_i = y_i^*$, otherwise y_i is censored so that $y_i < y_i^*$. We are also given binary censorship labels l_i , so that $l_i = 1$ if y_i is censored and $l_i = 0$ otherwise (e.g., labels could be recovered by comparing observed demand to available supply).

Tobit parameterizes the dependency of y_i^* on explanatory features $\mathbf{x}_i$ through

a linear relationship with parameters β and noise term ε_i , where all ε_i are independently and normally distributed with mean zero and variance σ^2 , namely:

$$y_i^* = \beta^T \mathbf{x}_t + \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2). \quad (3.1)$$

There are multiple variations of the Tobit model depending on where and when censoring arises. In this work, without loss of generality, we deal with upper censorship, also known as *Type I*, where y_i is upper-bounded by a given threshold y_u , so that:

$$y_i = \begin{cases} y_i^*, & \text{if } y_i^* < y_u \\ y_u, & \text{if } y_i^* \geq y_u \end{cases}. \quad (3.2)$$

The likelihood function in this case can be derived from (3.1) and (3.2), as follows.

1. If $l_i = 0$, then y_i is non-censored and so its likelihood is:

$$\frac{1}{\sigma} \phi \left(\frac{y_i - \beta^T \mathbf{x}_t}{\sigma} \right), \quad (3.3)$$

where ϕ is the standard Gaussian probability density function.

2. Otherwise, i.e., if $l_i = 1$, then y_i is censored and so its likelihood is:

$$1 - \Phi \left(\frac{y_i - \beta^T \mathbf{x}_t}{\sigma} \right), \quad (3.4)$$

where Φ is the standard Gaussian cumulative density function.

Because all observations are assumed to be independent, their joint likelihood is:

$$\prod_i \left\{ \frac{1}{\sigma} \phi \left(\frac{y_i - \beta^T \mathbf{x}_t}{\sigma} \right) \right\}^{1-l_i} \left\{ 1 - \Phi \left(\frac{y_i - \beta^T \mathbf{x}_t}{\sigma} \right) \right\}^{l_i}, \quad (3.5)$$

which is a function of β and σ .

3.3.2 Gaussian Processes

Gaussian Processes (GPs) (Rasmussen and Williams, 2005) are an extremely powerful and flexible tool belonging to the field of probabilistic machine learning (Ghahramani, 2015). GPs have been applied successfully to both classification and regression tasks regarding various transport related scenarios such as travel times (Ide and Kato, 2009, Rodrigues et al., 2017), congestion models (Liu et al., 2013), crowdsourced data (Rodrigues et al., 2018, Rodrigues and Pereira, 2018),

traffic volumes (Xie et al., 2010), etc. For example, given a finite set of points for regression, there are typically infinitely many functions which fit the data, and GPs offer an elegant approach to this problem by assigning a probability to every possible function. Moreover, GPs implicitly adopt a full probabilistic approach, thus enabling the structured quantification of the confidence – or equivalently, the uncertainty – in the predictions of a GP model. This ease in uncertainty quantification is one of the principal reasons why we chose to use GPs for demand prediction in the shared mobility domain. Indeed, transport service providers are not only interested in more accurate demand models, but also, and maybe most importantly, wish to make operational decisions based on the measure with which the model is confident of its predictions.

Given a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ with n input vectors $\mathbf{x}_i$ and scalar outputs y_i , the goal of GPs is to learn the underlying data distribution by defining a distribution over functions. GPs model this distribution by placing a multivariate Gaussian distribution defined by a *mean function* $m(\mathbf{x})$ and a *covariance function* $k(\mathbf{x}, \mathbf{x}')$ (between all possible pairwise combinations of input points $\{\mathbf{x}, \mathbf{x}'\}$ in the dataset), or *kernel*, on a latent variable $\mathbf{f}$. More concretely, a GP can be seen as a collection of random variables which have a joint Gaussian distribution. GPs are therefore a Bayesian approach which assumes a priori that function values behave according to:

$$p(\mathbf{f}|\mathbf{x}_1, \dots, \mathbf{x}_n) = \mathcal{N}(\mathbf{m}, \mathbf{K}), \quad (3.6)$$

where $\mathbf{f} = [f_1, \dots, f_n]^T$ is the vector of latent random variables, $\mathbf{m}$ is a mean vector, and $\mathbf{K}$ is a covariance matrix with entries defined by a covariance function k , so that $\mathbf{K}_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$. As is customary in GP modeling, we shall assume without loss of generality that the joint Gaussian distribution is centered on $\mathbf{m} \equiv \mathbf{0}$. Also, because the response variable is continuous, we model each y_i as generated from a Gaussian distribution centered on f_i with noise variance σ^2 .

As the resources needed for training GPs depend on the size of the train set, GPs have traditionally been limited to several thousands of training points. Nevertheless, GP scalability is an active field of research, and several methods have been devised for applying GPs to large datasets, including large-scale time series. For instance, GPs have been scaled to millions of observations through approximate inference methods, such as variational inference (Hensman et al., 2013) and sparse regression (Titsias, 2009). Furthermore, exact inference of GPs has recently been achieved on a million data points via parallelization on multiple Graphical Processing Units (Wang et al., 2019).

3.3.3 Kernels

A fundamental step in Gaussian Process modeling is the choice of the covariance matrix $\mathbf{K}$. This not only describes the shape of the underlying multivariate Gaussian distribution, but also determines the characteristics of the function we want to model. Intuitively, because entry $\mathbf{K}_{i,j}$ defines the correlation between the i -th and the j -th random variables, the kernel describes a measure of similarity between points, ultimately controlling the shape that the GP function distribution adopts.

Multiple kernels can be combined, e.g., by addition and multiplication, to generate more complex covariance functions, thus allowing for a great flexibility and better encoding of domain knowledge in the regression model. For the datasets in our experiments (Section 3.4), we use different combinations of the following three kernels, where $l_2(\mathbf{x} - \mathbf{x}')$ denotes the Euclidean distance between vectors $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$:

1. *Squared Exponential Kernel (SE)*

$$k_{SE}(\mathbf{x}, \mathbf{x}') = \lambda^2 \exp\left(-\frac{(l_2(\mathbf{x} - \mathbf{x}'))^2}{2\tau^2}\right) \quad (3.7)$$

The SE kernel (also called RBF: Radial Basic Function) intuitively encodes the idea that nearby points should be correlated, therefore generating relatively smooth functions. The kernel is parameterized by variance λ^2 and length scale τ : for larger τ , farther points are considered more similar; λ acts as an additional scaling factor, so that larger λ corresponds to higher spread of functions around the mean.

2. *Periodic Kernel*

$$k_{Per}(\mathbf{x}, \mathbf{x}') = \lambda^2 \exp\left(-\frac{2 \sin^2(\pi l_2(\mathbf{x} - \mathbf{x}') / \rho)}{\tau^2}\right) \quad (3.8)$$

The Periodic kernel allows modeling functions with repeated patterns, and can thus extrapolate seasonality in time-series data. Parameters τ and λ^2 have the same effect as for the SE kernel, while parameter ρ corresponds to period length of patterns.

3. *Matérn Kernel*

$$k_{Mat}(\mathbf{x}, \mathbf{x}') = \lambda^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} \frac{l_2(\mathbf{x} - \mathbf{x}')}{\tau}\right)^\nu K_\nu\left(\sqrt{2\nu} \frac{l_2(\mathbf{x} - \mathbf{x}')}{\tau}\right), \quad (3.9)$$

where Γ is the Gamma function, K_ν is the Bessel function (Abramowitz and Stegun, 1965), and as before, τ is lengthscale and λ is variance. The Matérn

kernel can be considered as a generalization of the SE kernel, parameterized by positive parameters ν and τ , where lower values of ν yield less smooth functions.

As is common, we select kernel parameters in our experiments through Type-II Maximum Likelihood, also known as Empirical Bayes, whereby latent variables are integrated out.

3.3.4 Gaussian Process with Censored Likelihood

We have so far presented two separate modeling approaches: Tobit models for censored data, and non-parametric Gaussian Processes for flexible regression of arbitrarily complex patterns. For non-parametric censored regression, we now combine the two approaches within one model by defining a censorship-aware likelihood function for GPs:

$$\prod_i \left\{ \frac{1}{\sigma} \phi \left(\frac{y_i - f_i}{\sigma} \right) \right\}^{1-l_i} \left\{ 1 - \Phi \left(\frac{y_i - f_i}{\sigma} \right) \right\}^{l_i}. \quad (3.10)$$

(3.10) is obtained from (3.5) by replacing $\beta^T \mathbf{x}_t$, the Tobit prediction, with f_i , the GP prediction. This formulation thus extends the likelihood defined in (Groot and Lucas 2012) by allowing for varying thresholds through the use of censorship labels l_i . We next propose an inference procedure for obtaining the posterior distribution of Censored GP with this likelihood function.

3.3.4.1 Inference

In a Bayesian setting, we are interested in computing the posterior over the latent $\mathbf{f}$, given the response $\mathbf{y}$ and features $\mathbf{X}$. Bayes' rule defines this posterior exactly:

$$p(\mathbf{f}|\mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{f})p(\mathbf{y}|\mathbf{f}, \mathbf{X})}{p(\mathbf{y}|\mathbf{X})}. \quad (3.11)$$

However, exact calculation of the denominator in (3.11) is intractable because of the Censored GP likelihood ((3.10)), as also explained in (Rasmussen and Williams, 2005). The posterior distribution thus needs to be estimated approximately, which we do in this study via *Expectation Propagation* (EP) (Minka, 2001). EP alleviates the denominator intractability by approximating the likelihood of each observation through an un-normalized Gaussian in the latent f_i , as:

$$p(y_i|f_i) \simeq t_i(f_i|\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2) \triangleq \tilde{Z}_i \mathcal{N}(f_i|\tilde{\mu}_i, \tilde{\sigma}_i^2), \quad (3.12)$$

where t_i is the i -th factor, or site, with site parameters $\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2$. The likelihood is then approximated as a product of the n independent local likelihood approximations:

$$\prod_{i=1}^n t_i(f_i | \tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2) = \mathcal{N}(\tilde{\mu}, \tilde{\Sigma}) \prod_{i=1}^n \tilde{Z}_i, \quad (3.13)$$

where $\tilde{\mu} = [\tilde{\mu}_1, \dots, \tilde{\mu}_n]$, and $\tilde{\Sigma}$ is a diagonal covariance matrix with $\tilde{\Sigma}_{ii} = \tilde{\sigma}_i^2$. Consequently, the posterior $p(\mathbf{f} | \mathbf{y}, \mathbf{X})$ is approximated by

$$q(\mathbf{f} | \mathbf{y}, \mathbf{X}) \triangleq \frac{1}{Z_{EP}} p(\mathbf{f} | \mathbf{X}) \prod_{i=1}^n t_i(f_i | \tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2) = \mathcal{N}(\mu, \Sigma), \quad (3.14)$$

where $\Sigma = (\mathbf{K}^{-1} + \tilde{\Sigma}^{-1})^{-1}$, $\mu = \Sigma \tilde{\Sigma}^{-1} \tilde{\mu}$, and Z_{EP} is the EP approximation to the marginal likelihood.

The key idea in EP is to update the single site approximation t_i sequentially by iterating the following four steps. Firstly, given a current approximation of the posterior, the current t_i is left out from this approximation, defining a *cavity distribution*:

$$q_{-i}(f_i) \propto \int p(\mathbf{f} | \mathbf{X}) \prod_{j \neq i} t_j(f_j | \tilde{Z}_j, \tilde{\mu}_j, \tilde{\sigma}_j^2) df_j, \quad (3.15)$$

$$q_{-i}(f_i) \triangleq \mathcal{N}(f_i | \mu_{-i}, \sigma_{-i}^2),$$

where $\mu_{-i} = \sigma_{-i}^2 (\sigma_i^{-2} \mu_i - \tilde{\sigma}_i^{-2} \tilde{\mu}_i)$ and $\sigma_{-i}^2 = (\sigma_i^{-2} - \tilde{\sigma}_i^{-2})^{-1}$.

The second step is the computation of the target non-Gaussian marginal combining the cavity distribution with the exact likelihood $p(y_i | f_i)$:

$$q_{-i}(f_i) p(y_i | f_i). \quad (3.16)$$

Thirdly, a Gaussian approximation to the non-Gaussian marginal $\hat{q}(f_i)$ is chosen such that it best approximates the product of the cavity distribution and the exact likelihood:

$$\hat{q}(f_i) \triangleq \hat{Z}_i \mathcal{N}(\hat{\mu}_i, \hat{\sigma}_i^2) \simeq q_{-i}(f_i) p(y_i | f_i). \quad (3.17)$$

To achieve this, EP uses the property that if $\hat{q}(f_i)$ is Gaussian, then the distribution which minimizes the Kullback-Leibler divergence $\text{KL}(q_{-i}(f_i) p(y_i | f_i) \| \hat{q}(f_i))$, is the distribution whose first and second moments match the corresponding moments of $q_{-i}(f_i) p(y_i | f_i)$. In addition, given that $\hat{q}(f_i)$ is un-normalized, EP also imposes the condition that the zero-th moment should match. Formally then, EP minimises the following objective:

$$\text{KL}(q_{-i}(f_i) p(y_i | f_i) \| \hat{q}(f_i)) = \mathbb{E}_{q_{-i}(f_i) p(y_i | f_i)} \left[\log \left(\frac{q_{-i}(f_i) p(y_i | f_i)}{\hat{q}(f_i)} \right) \right]. \quad (3.18)$$

Lastly, EP chooses the t_i for which the posterior has the marginal from the third step.

3.3.4.2 Censored Likelihood Moments

The implementation of EP in our experiments is based on GPpy¹, an open source Gaussian Processes package in Python. We have implemented the censored likelihood and defined the following moments for the EP inference procedure:

$$\begin{aligned}\hat{Z}_i &= \int q_{-i}(f_i) t_i df_i, \\ \hat{\mu}_i &= \mathbb{E}_q[f_i], \\ \hat{\sigma}_i &= \mathbb{E}_q[(f_i - \mathbb{E}_q[f_i])^2].\end{aligned}\tag{3.19}$$

The moments of our censored likelihood ((3.10)) can be defined analytically, depending on the censoring upper limit y_u , as follows (for a detailed derivation, see (Groot and Lucas, 2012)).

- If $y_i < y_u$, then:

$$\begin{aligned}\hat{Z}_i &= \frac{1}{\sqrt{2\pi(\sigma^2 + \sigma_{-i}^2)}} \exp\left(-\frac{(y_i - \mu_{-i})^2}{2(\sigma^2 + \sigma_{-i}^2)}\right), \\ \hat{\mu}_i &= \mu_{-i} + \sigma_{-i}^2 \left(\frac{y_i - \mu_{-i}}{\sigma^2 + \sigma_{-i}^2}\right), \\ \hat{\sigma}_i &= \sigma_{-i}^2 - \sigma_{-i}^4 \left(\frac{1}{(\sigma^2 + \sigma_{-i}^2)}\right).\end{aligned}\tag{3.20}$$

- Otherwise, $y_i = y_u$, and:

$$\begin{aligned}\hat{Z}_i &= \Phi(\bar{z}_i), \\ \hat{\mu}_i &= \mu_{-i} + \frac{\sigma_{-i}^2 \mathcal{N}(\bar{z}_i)}{\Phi(\bar{z}_i) \sqrt{\sigma^2 + \sigma_{-i}^2}}, \\ \hat{\sigma}_i &= \sigma_{-i}^2 - \frac{\sigma_{-i}^4 \mathcal{N}(\bar{z}_i)}{\Phi(\bar{z}_i)(\sigma^2 + \sigma_{-i}^2)} \left(\bar{z}_i + \frac{\mathcal{N}(\bar{z}_i)}{\Phi(\bar{z}_i)}\right),\end{aligned}\tag{3.21}$$

where Φ is the Gaussian Cumulative Density Function, and

$$\bar{z}_i = (\mu_{-i} - y_u) / \sqrt{\sigma^2 + \sigma_{-i}^2}.\tag{3.22}$$

3.4 Experiments

In this Section, we execute experiments for several datasets as follows. First, to convey a high-level intuition on how censored models can have an advantage

¹<https://sheffielddml.github.io/GPy/>

compared to non-censored models, we start with two relatively simple datasets: synthetically generated data in Section 3.4.2, and real-world motorcycle accident data in Section 3.4.3. Equipped with this intuition, we then proceed to more elaborate real-world datasets. In Section 3.4.4, we then build models to predict bike pickups for a major bike-sharing service provider in Copenhagen, Denmark. Then in Section 3.4.5, we use data about yellow taxis in New York City. For both bike-sharing and taxis, pickups thus represent mobility demand, which is affected by the censoring phenomenon introduced in previous sections because of finite supply.

3.4.1 Models

As introduced in previous sections, the focus of our experiments² is the comparison between *Censored* and *Non-Censored* models in the estimation of true demand patterns. We thus compare three GP models:

- (i) *Non-Censored Gaussian Process (NCGP)*: represents the Gaussian Process model most commonly used in literature, i.e., with Gaussian observation likelihood. NCGP is trained on the entire dataset, consisting of both censored and non-censored observations, without discerning between them.
- (ii) *Non-Censored Gaussian Process, Aware of censorship (NCGP-A)*: functionally equivalent to NCGP, but uses information on censoring as a pre-processing step. That is, NCGP-A is trained only on non-censored points, thus avoiding exposure to a biased version of the true demand (because of censoring). This, however, comes at the cost of ignoring relevant information embedded in the censored data
- (iii) *Censored Gaussian Process (CGP)*: this model considers all observations – censored and non-censored – through the likelihood function defined in (3.10) (Section 3.3.4).

3.4.2 Synthetic Dataset

To intuitively illustrate the fundamental differences between the models, let us first show how they behave when fitted plainly with an SE kernel and hyperparameters $\lambda = \tau = 1$. For this, we use a synthetically generated dataset, where the censoring pattern has a relevant and meaningful structure. We define a

²Source code available at: <https://github.com/DanieleGammelli/CensoredGP>

Figure 3.1: Synthetically generated data (a) and model fits for NCGP (b), NCGP-A (c) and CGP (d). The dashed line is the latent true function f^* ; crosses and circles are, respectively, censored and non-censored observations; the continuous line and the shaded area are, respectively, the posterior mean over the GP latent variable $\mathbf{f}$ and the corresponding posterior 95% confidence interval.

latent function for $x \in \mathbb{R}$:

$$f^*(x) = 2 + \frac{\sin(2x)}{2} + \frac{x}{10}, \quad (3.23)$$

which we wish to recover from censored, noisy observations.

We generate these observations in two steps. First, for 100 equally spaced points x_i in $[0, 10]$, we evaluate f^* and add noise independently sampled from $\mathcal{N}(0, 0.1)$. Then, we censor the points near the peaks of f^* (52% of all points) so that each censored observation is within 80% to 70% of the corresponding true value of f^* . In practice, we obtain this by independently and uniformly sampling censoring intensities $p_{c,i} \sim \mathcal{U}(0.2, 0.3)$ and consequently computing the censored observations $y_i = f_i^*(1 - p_{c,i})$. As seen in Figure 3.1a, this censoring process generates a cloud of points far from the true dynamics of the latent f^* near the peaks of the sinusoidal oscillation.

We then fit NCGP, NCGP-A and CGP to the observations and measure how well each of them recovers the true f^* . The resulting predictions by the three models in Figures 3.1b to 3.1d highlight interesting aspects worth underlying:

Figure 3.2: NCGP fit for intensifying censorship of the motorcycle accident data.

- Figure 3.1b on the previous page: NCGP predicts an approximately linear trend for observed data, which, if we did not know the true function behind our observations, would seem like a reasonable conclusion. Nevertheless, the predictions are definitely far from the true data generating process.
- Figure 3.1c on the preceding page: By discarding the censored observations, NCGP-A is able to correctly assess the behavior of the underlying function in regions close to observable data. However, NCGP-A is not able to generalize to regions affected by censoring.
- Figure 3.1d on the previous page: CGP, on the other hand is able to exploit its knowledge of censoring to make better sense of observable data. Through the use of a censored likelihood ((3.10)), CGP assigns higher probability to plausible functional patterns, which are coherent not only with the observable data but also with the censored function.

3.4.3 Real-World Dataset 1: Motorcycle Accident

Having demonstrated our approach on an artificial dataset, we now make use of the openly accessible Motorcycle Accident dataset (Silverman, 1985) used in literature for a wide variety of statistical learning tasks. The data consists of acceleration measurements over 133 consecutive ms. For this experiment, we

Figure 3.3: NCGP-A fit for intensifying censorship of the motorcycle accident data.

Figure 3.4: CGP fit for intensifying censorship of the motorcycle accident data.

aim at exploring how variations in the severity of the censoring process can impact the models' performance. In particular, we will be focusing on two

axes of variation: the *percentage* of censored points, which controls how often the true function is observable, and the *intensity* of censoring, which controls how far censored observations are from true values. Concretely, we apply the following censoring process in the experiments with Motorcycle data $y_1^*, \dots, y_n^*$, given any $0 \leq p \leq 1$ and $0 \leq a < b \leq 1$.

1. Initialize $y_i = y_i^*$ for all $i = 1..n$.
2. Define the set of censored observations N_c by randomly selecting $\lceil pn \rceil$ elements from $1..n$.
3. For each $i \in N_c$, independently sample a censorship intensity as $p_c^{(i)} \sim \mathcal{U}[a, b]$, and censor the i 'th observation as $y_i = \left(1 - p_c^{(i)}\right) y_i^*$.

We explore moderate to extreme censoring through increasing percentage of censored points, as $p = 0.1, 0.5, 0.9$, and increasing censorship intensity, as $[a, b] = [0, 0.33], [0.33, 0.66], [0.66, 1.0]$. Figures 3.2 to 3.4 on pages 54–55 show how each of NCGP, NCGP-A and CGP, respectively, reacts to these variations in censorship intensity: the percentage of censored observations increases from top to bottom, and censorship intensity increases from left to right. In each plot, the horizontal and vertical axes correspond to time and acceleration, respectively; we omit axis labels to reduce visual clutter, as also the specifics of this dataset are less important in the context of this experiment. We see that:

- Figure 3.2: Despite being able to successfully recover the latent function for small censorship intensities (first row), NCGP is then exposed to an excessive amount of bias because of the increased censoring. This results in completely distorted predictions, as the remaining non-censored points are essentially considered as outliers by the model.
- Figure 3.3: On the other hand, NCGP-A manages to avoid the bias coming from censored points by ignoring them, as long as the non-censored observations characterize well the latent function. When this condition no longer holds, NCGP-A is unable to generalize to unobserved domains, so that it yields biased predictions with high uncertainty.
- Figure 3.4: Lastly, CGP not only manages to avoid the bias introduced by censored observations in moderate to high censoring scenarios, but is also able to couple information coming from both censored and non censored observations to achieve better predictions in regions of extreme censoring.

Figure 3.5: Map with: 1) marked locations of Donkey Republic hubs, 2) the three super-hubs in our experiments, as big circles around constituent hubs.

3.4.4 Real-World Dataset 2: Bike-Sharing Demand and Supply

In this Section, we deal with the problem of building a demand prediction model for a bike-sharing system. We use real-world data provided by Donkey Republic, a major bike-sharing provider in the city of Copenhagen, Denmark. Donkey Republic can be considered a hub-based service, meaning that the user of the service is not free to pick up or drop off a bike in any location, but is restricted to a certain number of virtual hubs around Copenhagen. Our objective is to model daily rental demand in the hub network.

3.4.4.1 Data Aggregation

The given data consists of individual records of users renting and returning bikes in hubs during 379 days: from 1 March 2018 until 14 March 2019. Hence before modeling daily rentals, we aggregate the data both spatially and temporally. Spatially, 32 hubs were aggregated in three *super-hubs* by selecting three main service areas (such as the central station and main tourist attractions) and considering a 200 m radius around these points of interest (Figure 3.5). The choice of radius is justified by Donkey Republic business insights, whereby users typically agree to walk at most 200 m to rent a bike and typically give up if the bike is farther. Temporally, the data at our disposal allowed us to retrieve the time-series of daily rental pickups regarding the three super-hubs, which will represent the target of our prediction model.

The spatial aggregation of individual hubs into super-hubs is an important modeling step in building the prediction model, for two main reasons:

1. Time-series for individual hubs reveal excessive fluctuations in rental behavior. Hence separate treatment of individual hubs could likely hide most of the valuable regularities in the data and ultimately expose the predictive models to an excessive amount of noise.
2. As seen in Figure 3.5 on the previous page, the individual hubs are very close one to the other, especially in central areas of the city. Demand patterns between neighboring hubs can thus be well correlated, and a bike-sharing provider would actually benefit more from understanding the demand in an entire area rather than in a single hub. Moreover, bike-sharing demand conceptually covers an entire area, as bike-sharing users would likely walk a few dozen meters more to rent a bike. Hence from here on, we assume that if no bikes are available at some hub, then users who wish to rent a bike are willing to walk to any other hub in the same super-hub.

3.4.4.2 Censorship of Bike-Sharing Data

Ideally and before modeling, we would like to have access to the true bike-sharing demand, free of any real-world censorship. However, this ideal setting is impossible, as historical data records are necessarily censored intrinsically to some extent. Consequently and for the sake of experimentation, we assume that the given historical data represents true demand (which is what ideally we would like to predict). This further allows us to censor the data manually and examine the effects of such censorship.

We apply manual censorship to the time series of each super-hub in two stages. In the first stage, for each day i in $N = \{1 \dots 379\}$, we let $\delta_i \in \{0, 1\}$ indicate whether at any moment during i there were no bikes available in the entire super-hub, and define accordingly the set of censored and non-censored observations:

$$N_c = \{i \in 1..379 : \delta_i = 1\}, \quad (3.24)$$

$$N_{nc} = \{i \in 1..379 : \delta_i = 0\} = N - N_c. \quad (3.25)$$

We then fix binary censorship labels as follows: $l_i = 1$ for $i \in N_c$ and $l_i = 0$ for $i \in N_{nc}$. The reason for doing so is that for every day in N_c , there was a moment with zero bike availability, and so there may have been additional demand, which the service could not satisfy and which was thus not recorded.

Having fixed the censorship labels, the second stage of censorship can be executed multiple times for different censorship intensities. That is, given a censorship intensity $0 \leq c \leq 1$, we censor each observation for which $l_i = 1$ to $(1 - c)$

Figure 3.6: Super-hub 3 data with 50% Censorship Intensity.

of its original value. For example, Figure 3.6 shows true demand (red) as well as the result of censoring it to 50% of its original value (grey) for each day in N_c (blue markers).

3.4.4.3 Modeling and Evaluation

As for previous experiments, we train models on the manually censored demand and then evaluate them on the observations without censorship – this is the measure of actual interest, as it represents true demand. Evaluation is done by comparing the posterior mean to the predicted mean through two commonly used measures: coefficient of determination (R^2) and Rooted Mean Squared Error (RMSE):

$$R^2 = 1 - \frac{\sum_{i=1}^m (\hat{\mu}_i - y_i^*)^2}{\sum_{i=1}^m (\bar{y}^* - y_i^*)^2}, \quad (3.26)$$

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^m (\hat{\mu}_i - y_i^*)^2}, \quad (3.27)$$

where $y_1^*, \dots, y_m^*$ are the true demand, $\hat{\mu}_1, \dots, \hat{\mu}_m$ are the corresponding evaluations of the posterior mean, and

$$\bar{y}^* = \frac{1}{n} \sum_{i=1}^n y_i^* \quad (3.28)$$

is the mean true demand.

As defined in Section 3.4.1, the three models are NCGP, NCGP-A and CGP. We equip each model with a kernel that consists of a combination of SE and Periodic kernels on the temporal index feature and a Matérn kernel on weather data. Namely, the covariance function between the i 'th and the j 'th sample is:

$$k(\mathbf{x}_i, \mathbf{x}_j) = k_{SE}(\mathbf{x}_{i,t}, \mathbf{x}_{j,t}) + k_{Per}(\mathbf{x}_{i,t}, \mathbf{x}_{j,t}) + k_{Mat}(\mathbf{x}_{i,w}, \mathbf{x}_{j,w}), \quad (3.29)$$

Solar Irradiance			Wind	Air	Rain
Global	short	wave	Direction min. [°]	Temperature [°C]	Fall [mm]
[W/m ²]					
Short	wave	horizontal	Direction max. [°]	Relative humidity [%]	Duration [s]
[W/m ²]					
Short	wave	direct normal	Direction avg. [°]	Pressure [hPa]	Intensity [mm/h]
[W/m ²]					
Downwelling	horizontal		Speed avg. [m/s]		
long wave [W/m ²]					
			Speed max. [m/s]		

Table 3.1: Weather features for bike-sharing data, collected in Copenhagen by (DTU Civil Engineering, 2019).

where $\mathbf{x}_{i,t} = [t]$ is the temporal index, and $\mathbf{x}_{i,w}$ consists of weather features as detailed in Table 3.1. Concretely, we first select the initial hyper-parameter values through prior predictive checks (Gelman et al., 2004), then estimate kernel parameters through Type-II Maximum Likelihood Estimation using Adam (for CGP) and BFGS (for NCGP and NCGP-A) to perform gradient ascent. This choice of optimization routines was the most performing in practice for each respective model.

The main goal of our experiments is to analyze how well these models can recover the *true* demand pattern after being trained on a censored version of the same demand. In particular, we are interested in investigating to what degree censored models (CGP) are able to recover the underlying demand pattern compared to standard regression models (NCGP, NCGP-A), and how this comparison evolves under different censorship intensities. Our experiments thus employ incremental censorship intensity $c = 0, 0.1, 0.2, \dots, 1.0$, so that c ranges between two extremes: from absence of censoring (*full observability* of the true demand) to complete censoring (no demand observed in historical data).

3.4.4.4 Results for Bike-Sharing Experiments

This section presents results for the predictive models implemented on each of the three time-series with cross-validation. Tables 3.4 to 3.6 in Section 3.6 detail the evaluation for each of the three super-hubs. We now concentrate on the results for super-hub 1, as presented in Figure 3.7 on the facing page, because they are representative also of the results for the other two super-hubs. The plots in Figure 3.7 on the next page are a visual representation of Table 3.4 on page 72 and compare the performances of NCGP, NCGP-A and CGP for different censorship intensities. We discern between evaluating model performance on the

Figure 3.7: Censoring performance analysis on the entire dataset and only on non-censored observations, regarding R^2 and RMSE for super-hub 1.

entire dataset (consisting of both censored and non-censored observations) vs. only on non-censored observations.

First, we compare the models that do not discard of any observations, namely, CGP and NCGP. Considering that a predictive model is better the more its RMSE is to close 0 and the more its R^2 is close to 1, the plots show that the two models are comparable under low degree of censoring. However, as the censorship intensifies, NCGP becomes strongly biased towards the censored observations, whereas CGP recovers the underlying demand much more consistently.

Next, we compare between NCGP-A and the CGP and see that NCGP-A achieves reasonable predictive accuracy, which is still mostly worse than the predictive accuracy of CGP. As outlined previously in Sections 3.4.2 and 3.4.3, NCGP-A accuracy depends highly on the extent to which observable data characterizes the full behavior of the latent function (in this case, true demand). Here, the percent of points affected by censoring falls between 20% and 40% for all the three super-hubs, so that NCGP-A has acceptable observability over the true demand. Even so, CGP outperforms NCGP-A also on just *non-censored* data; this suggests that using a censored likelihood not only allows models to avoid predictive bias on censored data, but also allows consistent understanding of the data generating process, ultimately leading to increased performance also on observable data.

In conclusion, and as outlined in previous Sections, the non-parametric nature of Censored GP allows it to effectively exploit the *concept* of censoring, thus preventing censored observations from biasing the entire demand model. In other words, Censored GP is capable of activating *censoring-awareness* depending on data only.

3.4.4.5 Behavior under Zero Censorship

It is worth emphasizing how Non-Censored GP is a special case of Censored model GP in those cases where all observations are assumed to be non-censored. This becomes evident when juxtaposing the two likelihood functions:

$$L_C = \prod_{i \in N_{nc}} \left(\frac{1}{\sigma} \phi \left(\frac{y_i - f_i}{\sigma} \right) \right) \prod_{i \in N_c} \left(1 - \Phi \left(\frac{y_i - f_i}{\sigma} \right) \right), \quad (3.30)$$

$$L_{NC} = \prod_{i \in N} \left(\frac{1}{\sigma} \phi \left(\frac{y_i - f_i}{\sigma} \right) \right), \quad (3.31)$$

which shows that in absence of censoring (i.e., when $N_c = \emptyset$), $L_C \equiv L_{NC}$.

For 0% censorship, we still see in Figure 3.7 on the previous page some difference in performance. To realize why this happens, recall that by our experimental design (Section 3.4.4.2), censorship labels l_i are pre-determined regardless of censorship intensity, hence some observations are labeled as censored even when censorship intensity is 0. Furthermore, the true demand in our experiments originates from historical data, which is itself intrinsically censored. It follows that the models are never exposed to utter absence of censored observations. Conversely, in a real-world scenario, perfect 0% censoring corresponds to complete absence of censored observations, so that censored and non-censored GP are equivalent.

3.4.5 Real-World Dataset 3: Taxi Demand and Supply

The previous Section dealt with *long-term* transport demand under *deterministic* censorship per available supply. In what follows, we complement this with a case study of *short-term* transport demand under *stochastic* censorship per available supply. To this end, we apply GP modeling to stochastically censored pickups of yellow, hail-based taxis (Donovan and Work, 2014) within an approximately $1 \text{ km} \times 1 \text{ km}$ squared area near LaGuardia airport in New York City. The true pickup counts, $\mathbf{y}^* = [y_1^*, \dots, y_{672}^*]$, are aggregated per 15 min consecutive intervals in the first week of June 2010. At that time, NYC Taxis were almost entirely of the yellow type and thus accounted for virtually all ride-hailing demand.

Algorithm 1: RandDropoff Censorship

input: $0 \leq \gamma \leq 1, 0 \leq c \leq 1, \mathbf{y}^* = [y_1^*, \dots, y_n^*]^T \in \mathbb{R}^n, \mathbf{d} = [d_0, \dots, d_{n-1}]^T \in \mathbb{R}^n$

1 foreach $i = 1..n$ do

2 Sample $l_i \in \{0, 1\}$ as $l_i \sim \text{Bernoulli}(p_{i,\gamma})$, where

$$p_{i,\gamma} = \left\{ 1 + \exp \left(\ln \left(\frac{1-\gamma}{\gamma} \right) - \frac{y_i - d_{i-1}}{y_i} \right) \right\}^{-1}. \quad (3.32)$$

3 $y_i \leftarrow y_i^* (1 - c)^{l_i}$.

4 end

5 return $\mathbf{y} = [y_1, \dots, y_n], \mathbf{k} = [l_1, \dots, l_n]$.

Figure 3.8: In RandDropoff, censorship probability $p_{i,\gamma}$ depends on a zero-intercept parameter γ and the difference between pickups y_i^* and dropoffs d_{i-1} . The higher γ is and the higher y_i^* is above d_{i-1} , the more likely will y_i^* be censored.

3.4.5.1 RandDropoff Experiments

Taxi demand is influenced by availability of vacant taxis. To approximate the number of vacant taxis, we use the number of dropoffs observed in the previous lag. That is, we use d_{i-1} , the observed total dropoffs in time step $i-1$, to stochastically censor y_i^* , the true pickups in time step i , as follows. First, we randomly select the censorship label $l_i \in \{0, 1\}$ with probability depending on y_i^*, d_{i-1} , and a parameter $\gamma \in (0, 1)$. As explained in Figure 3.8, approximately γ of all y_i^* are thus labeled as censored ($l_i = 1$). Next, we censor from above each observation for which $l_i = 1$, by setting y_i to $(1 - c)y_i^*$, where $c \in [0, 1]$ is a given parameter. c is thus the percent of pickups that are unobserved and so represents unsatisfied ride-hailing demand. This two-step procedure, named RandDropoff, is rigorously defined in Algorithm 1. Figure 3.9 on the following page illustrates an example of RandDropoff for one combination of γ, c .

We experiment with RandDropoff censorship for $\gamma = 0.1, 0.2, 0.3, 0.4$ and $c =$

Figure 3.9: Example of RandDropoff censorship with $\gamma = 0.1$ and $c = 0.5$.

$0, 0.1, \dots, 1$. For these values of γ , the average percentages of censored observations are, respectively, 13%, 23%, 32%, 40% – close to the expected values. We will later see that these values of γ suffice for reasoning about the behavior of each model as more observations become censored. The censorship magnitude increases with c , so that $c = 0$ and $c = 1$ represent limiting cases: when c tends to 0, censored observations become very close to true values, whereas when c tends to 1, censored observations are zeroed.

The models are again NCGP, NCGP-A and CGP, and fitting is done via BFGS with at most 1000 iterations, starting from the same initial hyper-parameters chosen through prior predictive checks. Because RandDropoff is stochastic, we experiment with each γ, c combination independently a total of 30 times. In each independent experiment, we fit and evaluate through cross-validation with 21 time-consecutive folds, each consisting of 32 observations over 8 hours. For each γ and c , we evaluate by comparing $\mathbf{y}$, the latent count of pickups, to the predicted means over all 30 experiments.

For all models, the explanatory features are $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_{672}]$, where for all time steps $t = 1 \dots 672$, $\mathbf{x}_t$ consists of: t , the corresponding hour-of-day in $0 \dots 23$, and the corresponding day-of-week in $0 \dots 6$ (this increases similarity between data points in the same day and improves fit accuracy). For NCGP-A and CGP, $\mathbf{x}_t$ also contains the binary censorship label l_t . Because weather conditions in the first week of June 2010 were quite stable, they do not serve here as features. The GP kernel for all three models is thus:

$$k(\mathbf{x}, \mathbf{x}') = k_{SE}(\mathbf{x}, \mathbf{x}') + k_{Per}(\mathbf{x}, \mathbf{x}') . \quad (3.33)$$

3.4.5.2 Results of RandDropoff Experiments

We evaluate the models both on the entire dataset and exclusively on non-censored observations. The results are illustrated in Figure 3.10 and detailed

(a) Evaluation over all observations.

(b) Evaluation over only non-censored observations.

Figure 3.10: Performance of models in RandDropoff experiments. Extreme values of NCGP are omitted for easier comparison between NCGP-A and CGP.

in Table 3.7 in Section 3.6. Next, we analyze the behavior of each model per Figure 3.10.

As γ and c increase, so do the percentage of censored observations and the intensity of censorship. Consequently, we see in Figure 3.10 that NCGP deteriorates rapidly as the censored observations draw it downwards, away from the latent pickups. Conversely, NCGP-A maintains a rather stable performance even as γ increases, which suggests that it has enough non-censored points for constructing a stable fit. Note that as NCGP-A ignores censored observations, its performance does not depend on c and thus traces a horizontal line for each γ .

For the limiting case of $\gamma = 0.4$ and $c = 0$, NCGP is the best performing mode,

slightly better than CGP. This, however, may be expected, because labeling many observations as censored without actually censoring them is misleading for any censorship-aware model, including CGP. In all other cases, CGP outperforms NCGP and NCGP-A, whether for the entire dataset or for only non-censored observations. This suggests that CGP is more reliable not only in coping with censorship, but also in situations where observations are known to accurately reflect the latent ground truth. Moreover, CGP performance follows a pattern, which becomes more pronounced as γ increases: CGP starts close to NCGP for $c = 0$, improves as c increases until about $c = 0.5$, then decreases and converges.

Finally, we note that we have also experimented with a similarly sized spatial area in the middle of Manhattan, NYC. Contrary to the time series used in this Section, the Manhattan cell exhibited a repetitive and regular demand pattern, for which NCGP-A and CGP performed quite closely. The noticeably better performance of CGP in this Section thus suggests that the advantages of censored modeling emerge in more challenging settings — where the ability to extract meaningful information from censored observations is indeed essential for capturing the underlying demand pattern.

3.5 Summary and Future Work

Building a model for demand prediction naturally relies on extrapolating knowledge from historical data. This is usually done by implementing different types of regression models, to both explain past demand behavior and compute reliable predictions for the future – a fundamental building block for a great number decision making processes. However, we have shown how a reliable predictive model must take into consideration *censoring*, especially in those cases in which demand is implicitly limited by supply. More importantly, we stressed the fact that, in the context of shared transport demand modeling, there is a need for models which can deal with censoring in a meaningful way, rather than resorting to different *data cleaning* techniques.

To deal with the censoring problem, we have constructed models that incorporate a *censored* likelihood function within a flexible, non-parametric Gaussian Process (GP). We compare this model to commonly used GP models, which incorporate a standard Gaussian likelihood, through a series of experiments on synthetic and real-world datasets. These experiments highlight how standard regression models are prone to return a biased model of demand under data censorship, whereas the proposed Censored GP model yields consistent predictions even under severe censorship.

The experimental results thus confirm the importance of censoring in demand modeling, especially in the transport scenario where demand and supply are naturally interdependent. More generally, our results support the idea of building more *knowledgeable* models instead of using case-dependent data cleaning techniques. This can be done by feeding the demand models insights on how the demand patterns actually behave, so that the models can adjust automatically to the available data.

For future work, we plan to study settings where censorship labels are partly or even completely unknown. We also note that shared mobility demand prediction can benefit from utilizing spatio-temporal correlations, as commutes between areas take place regularly and as nearby areas are likely to exhibit similar demand patterns. Hence also for future work, we plan to implement models which predict censored demand jointly for multiple areas.

Acknowledgement

The research leading to these results has received funding from the People Programme (Marie Curie Actions) of the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Individual Fellowship H2020-MSCA-IF-2016, ID number 745673.

3.6 Supplementary Results

3.6.1 Matrix-based Imputation

In response to a reviewer comment, we have also experimented and compared our proposed models with imputation based on matrix factorization. While the results of these additional experiments further supported our main message – that preserving censorship information can often benefit modeling – imputation was nevertheless out of the scope for this work. We next present the additional experiments and then explain why they were left out of the published work.

For matrix factorization-based imputation, we use Probabilistic Principal Component Analysis (PPCA) (Tipping and Bishop, 1999). A stochastic variant of the classic PCA algorithm (Wold et al., 1987), PPCA uses Gaussians as marginals for all latent elements, including any missing values in the given matrix. For each censored dataset in our work, we apply PPCA with latent space dimension 4, as suggested in (Rodrigues et al., 2018).

Figure 3.11: CGP (left) vs. NCGP-I (right) fit on the synthetic dataset.

As input for PPCA, we convert any vector $\mathbf{y}_{obs}$ of N observations to a matrix, as follows. First, we obtain $\mathbf{y}'_{obs}$ by replacing the censored observations in $\mathbf{y}_{obs}$ with missing values (NaN's). Next, we shift $\mathbf{y}'_{obs}$ to obtain lagged vectors $\mathbf{y}'_{obs,-3}, \mathbf{y}'_{obs,-2}, \dots, \mathbf{y}'_{obs,+3}$. Finally, we reshape each of $\mathbf{y}'_{obs}, \mathbf{y}'_{obs,-3}, \dots, \mathbf{y}'_{obs,+3}$ as a 7-by- $N/7$ matrix (7 divides N for all our datasets), and stack the matrices as input to PPCA. When PPCA is done, we extract the imputed observations $\mathbf{y}_{imp}$ from the matrix rows that correspond to $\mathbf{y}'_{obs}$.

Having thus obtained the imputed observations, we next fit and evaluate a Gaussian Process (GP) on $\mathbf{y}_{imp}$, precisely as we do for NCGP on $\mathbf{y}_{obs}$. We denote the GP on $\mathbf{y}_{imp}$ as **NCGP-I**: Non-Censored Gaussian Process with Imputation. Following is an overview of the results.

For the synthetic dataset in Section 3.4.2, Figure 3.11 shows that NCGP-I captures the sinusoidal trend, yet yields a mean function which is far from the actual, latent function. This happens because the mean function in NCGP-I follows the imputed information, which in turn relies on a set of non-censored observations that are far from the actual signal. In contrast, by preserving the information in the censored observations too, CGP yields a mean function which is much closer to actual, underlying function.

For the motorcycle incident dataset in Section 3.4.3, Figure 3.12 on the facing page shows how the ability of NCGP-I to capture the latent dynamics is highly dependent on the difficulty of the imputation task, which could in turn act as a bottleneck for performance. In particular, as also discussed in Section 3.4.3, the grid in Figure 3.12 qualitatively assesses performance for variations of both the percentage of censored observations and censoring intensity. Figure 3.12 clearly highlights how, for regimes of censoring in which the underlying function is well described by available observations (i.e., low percentage of censored observations), NCGP-I is able to obtain results comparable with CGP. On the other hand, regimes of high percentage of censored points (i.e., last row in Figure 3.12), represent a challenging scenario for imputation methods, causing therefore NCGP-I to fail in representing the underlying functional form.

Figure 3.12: NCGP-I fit on the motorcycle dataset.

Figure 3.13: Censoring performance analysis, with NCGP-I, for bike-sharing data.

For the real-world bike sharing dataset in Section 3.4.4, Figure 3.13 and Table 3.2 on the following page show that in MAE, RMSE and R^2 terms, NCGP-I performs very closely to NCGP-A (which, at test time, can be considered as indirectly imputing censored observations by discarding them at training time). Hence for this dataset, imputing the censored observations is nearly as good as simply removing them. As before, CGP is still the best performing model in all cases, except for extreme censoring intensities.

For the real-world taxi dataset in Section 3.4.5, Table 3.3 shows that in Mean Absolute Error (MAE) terms, NCGP-I outperforms both NCGP and NCGP-A. However, CGP is still the best performing model in all cases, except for the two lowest censorship magnitudes. We also note that counter-intuitively, NCGP-I performs better as the percent of censored observations increases. Closer examination reveals that for higher γ , PPCA imputation tends to increasingly over- and under-estimate the missing observations, which in turn can bring the NCGP-I mean function closer to the irregular latent signal in this dataset.

Censorship Intensity:		0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
MAE	NCGP	6.23	6.21	6.30	6.56	7.02	7.48	8.05	8.66	9.30	9.99	10.71
	NCGP-A	6.84	6.84	6.84	6.84	6.84	6.84	6.84	6.84	6.84	6.84	6.84
	NCGP-I	6.96	6.96	6.96	6.96	6.96	6.96	6.96	6.96	6.96	6.96	6.96
	CGP	6.74	6.49	6.38	6.25	6.25	6.30	6.32	6.46	6.58	6.77	6.99
RMSE	NCGP	8.31	8.38	8.6	8.99	9.52	10.07	10.74	11.48	12.29	13.14	14.03
	NCGP-A	9.03	9.03	9.03	9.03	9.03	9.03	9.03	9.03	9.03	9.03	9.03
	NCGP-I	9.13	9.13	9.13	9.13	9.13	9.13	9.13	9.13	9.13	9.13	9.13
	CGP	8.73	8.47	8.38	8.26	8.34	8.44	8.51	8.67	8.85	9.04	9.28
R^2	NCGP	0.69	0.68	0.66	0.63	0.59	0.54	0.48	0.40	0.32	0.22	0.11
	NCGP-A	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63
	NCGP-I	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62
	CGP	0.65	0.68	0.68	0.69	0.69	0.68	0.67	0.66	0.65	0.63	0.61

Table 3.2: Model performance, with NCGP-I, for Entire Dataset of Bike Super-hub 1

$c :$		0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
MAE												
$\gamma = 0.1$	NCGP	22.29	22.74	23.04	23.27	22.81	22.53	22.50	22.43	22.53	22.82	23.25
	NCGP-A	23.87	23.87	23.87	23.87	23.87	23.87	23.87	23.87	23.87	23.87	23.87
	NCGP-I	22.64	22.64	22.64	22.64	22.64	22.64	22.64	22.64	22.64	22.64	22.64
	CGP	21.97	22.03	21.94	21.87	21.85	21.71	21.69	21.55	21.36	21.21	21.00
$\gamma = 0.2$	NCGP	22.28	22.79	23.32	23.39	23.16	23.10	23.26	23.80	24.62	25.57	26.65
	NCGP-A	23.43	23.43	23.43	23.43	23.43	23.43	23.43	23.43	23.43	23.43	23.43
	NCGP-I	22.25	22.25	22.25	22.25	22.25	22.25	22.25	22.25	22.25	22.25	22.25
	CGP	22.00	21.61	21.47	21.24	21.24	21.23	21.26	21.20	21.16	21.00	20.89
$\gamma = 0.3$	NCGP	22.29	22.74	23.23	23.24	23.13	23.61	24.54	25.77	27.17	28.73	30.29
	NCGP-A	23.10	23.10	23.10	23.10	23.10	23.10	23.10	23.10	23.10	23.10	23.10
	NCGP-I	21.52	21.52	21.52	21.52	21.52	21.52	21.52	21.52	21.52	21.52	21.52
	CGP	22.24	21.69	21.25	21.18	21.07	21.23	21.38	21.46	21.33	21.15	21.06
$\gamma = 0.4$	NCGP	22.29	22.96	23.59	23.57	23.93	24.63	25.99	27.86	30.01	32.14	34.54
	NCGP-A	23.26	23.26	23.26	23.26	23.26	23.26	23.26	23.26	23.26	23.26	23.26
	NCGP-I	21.49	21.49	21.49	21.49	21.49	21.49	21.49	21.49	21.49	21.49	21.49
	CGP	22.65	21.91	21.31	21.01	20.96	21.06	21.07	21.32	21.25	21.23	21.24
RMSE												
$\gamma = 0.1$	NCGP	31.29	31.72	31.96	32.17	31.97	31.98	32.18	32.31	32.53	32.90	33.49
	NCGP-A	32.87	32.87	32.87	32.87	32.87	32.87	32.87	32.87	32.87	32.87	32.87
	NCGP-I	31.85	31.85	31.85	31.85	31.85	31.85	31.85	31.85	31.85	31.85	31.85
	CGP	30.26	30.22	30.11	30.09	30.17	30.16	30.22	30.17	30.06	29.98	29.85
$\gamma = 0.2$	NCGP	31.29	31.86	32.35	32.67	32.84	33.06	33.51	34.39	35.53	36.74	38.08
	NCGP-A	32.55	32.55	32.55	32.55	32.55	32.55	32.55	32.55	32.55	32.55	32.55
	NCGP-I	31.47	31.47	31.47	31.47	31.47	31.47	31.47	31.47	31.47	31.47	31.47
	CGP	30.60	30.05	29.93	29.85	30.02	30.16	30.28	30.34	30.35	30.18	30.00
$\gamma = 0.3$	NCGP	31.29	31.86	32.36	32.79	33.02	33.96	35.34	36.90	38.70	40.62	42.59
	NCGP-A	32.44	32.44	32.44	32.44	32.44	32.44	32.44	32.44	32.44	32.44	32.44
	NCGP-I	30.63	30.63	30.63	30.63	30.63	30.63	30.63	30.63	30.63	30.63	30.63
	CGP	31.03	30.31	29.89	29.84	29.92	30.29	30.59	30.80	30.72	30.57	30.49
$\gamma = 0.4$	NCGP	31.30	32.07	32.77	33.28	34.14	35.37	37.18	39.40	41.95	44.68	47.58
	NCGP-A	32.91	32.91	32.91	32.91	32.91	32.91	32.91	32.91	32.91	32.91	32.91
	NCGP-I	30.48	30.48	30.48	30.48	30.48	30.48	30.48	30.48	30.48	30.48	30.48
	CGP	31.61	30.68	30.05	29.84	29.95	30.27	30.55	30.97	30.97	30.93	30.95
R^2												
$\gamma = 0.1$	NCGP	0.48	0.47	0.46	0.45	0.46	0.46	0.45	0.45	0.44	0.43	0.41
	NCGP-A	0.43	0.43	0.43	0.43	0.43	0.43	0.43	0.43	0.43	0.43	0.43
	NCGP-I	0.46	0.46	0.46	0.46	0.46	0.46	0.46	0.46	0.46	0.46	0.46
	CGP	0.52	0.52	0.52	0.52	0.52	0.52	0.52	0.52	0.52	0.53	0.53
$\gamma = 0.2$	NCGP	0.48	0.46	0.45	0.44	0.43	0.42	0.41	0.37	0.33	0.29	0.23
	NCGP-A	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44
	NCGP-I	0.48	0.48	0.48	0.48	0.48	0.48	0.48	0.48	0.48	0.48	0.48
	CGP	0.51	0.52	0.53	0.53	0.52	0.52	0.52	0.51	0.51	0.52	0.52
$\gamma = 0.3$	NCGP	0.48	0.46	0.45	0.43	0.42	0.39	0.34	0.28	0.21	0.13	0.04
	NCGP-A	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44
	NCGP-I	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
	CGP	0.49	0.51	0.53	0.53	0.53	0.52	0.51	0.50	0.50	0.51	0.51
$\gamma = 0.4$	NCGP	0.48	0.46	0.43	0.41	0.38	0.34	0.27	0.18	0.07	−0.06	−0.20
	NCGP-A	0.43	0.43	0.43	0.43	0.43	0.43	0.43	0.43	0.43	0.43	0.43
	NCGP-I	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51	0.51
	CGP	0.47	0.50	0.52	0.53	0.53	0.52	0.51	0.49	0.49	0.49	0.49

Table 3.3: Model Performance, with NCGP-I, for Entire Taxi Dataset

In summation, we have obtained that CGP still mostly outperforms the other, censorship-unaware models. We have also seen that the performance of NCGP-I is tied with the quality of imputation, and by extension, this should hold true for any imputation-based model. Hence in situations where accurate imputation is more challenging, the overall quality of an imputation-based model will also likely be affected. Conversely, censorship-aware models such as CGP can take advantage of knowledge preserved in partial observations – e.g., that censored data provides a lower bound on the underlying latent demand – when looking for the latent generative process that best accounts for the observed dynamics.

Although these additional experiments further support the main messages of our work, they still fall out of its scope. Imputation techniques are, by definition, designed to compensate for completely missing data, which is not the case in this study. Rather, we deal with Transport demand modeling, where data is characterized by partial observability over the latent process (i.e., latent demand) and is thus not only available, but also potentially information-rich. Consequently, expanding the work with imputation-based modeling would add extraneous details that hide its main contributions and might confuse readers. We have thus left these additional experiments out of the published work.

3.6.2 Additional Numeric Results

The following Tables provide additional numeric results for Section 3.4, with best values highlighted in bold: Tables 3.4 to 3.6 on the next page for Section 3.4.4, and Table 3.7 on page 73 for Section 3.4.5.

Censorship Intensity:		0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Entire Dataset												
RMSE	NCGP	8.31	8.38	8.6	8.99	9.52	10.07	10.74	11.48	12.29	13.14	14.03
	NCGP-A	9.03	9.03	9.03	9.03	9.03	9.03	9.03	9.03	9.03	9.03	9.03
	CGP	8.73	8.47	8.38	8.26	8.34	8.44	8.51	8.67	8.85	9.04	9.28
R^2	NCGP	0.69	0.68	0.66	0.63	0.59	0.54	0.48	0.40	0.32	0.22	0.11
	NCGP-A	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63
	CGP	0.65	0.68	0.68	0.69	0.69	0.68	0.67	0.66	0.65	0.63	0.61
Non-Censored Data												
RMSE	NCGP	7.96	7.99	8.11	8.36	8.74	9.02	9.43	9.91	10.44	11.01	11.61
	NCGP-A	8.73	8.73	8.73	8.73	8.73	8.73	8.73	8.73	8.73	8.73	8.73
	CGP	8.19	8.06	8.00	7.91	7.98	8.02	8.08	8.20	8.25	8.40	8.54
R^2	NCGP	0.69	0.68	0.67	0.65	0.62	0.60	0.56	0.51	0.46	0.40	0.33
	NCGP-A	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62
	CGP	0.67	0.68	0.68	0.69	0.68	0.68	0.68	0.67	0.66	0.65	0.64

Table 3.4: Model Performance for Super-hub 1

Censorship Intensity:		0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Entire Dataset												
RMSE	NCGP	6.17	6.22	6.37	6.63	7.00	7.42	7.91	8.46	9.05	9.67	10.33
	NCGP-A	7.28	7.28	7.28	7.28	7.28	7.28	7.28	7.28	7.28	7.28	7.28
	CGP	6.74	6.57	6.44	6.43	6.52	6.59	6.72	6.90	7.05	7.22	7.40
R^2	NCGP	0.65	0.65	0.63	0.60	0.56	0.50	0.43	0.35	0.26	0.15	0.03
	NCGP-A	0.52	0.52	0.52	0.52	0.52	0.52	0.52	0.52	0.52	0.52	0.52
	CGP	0.59	0.61	0.62	0.62	0.61	0.61	0.59	0.57	0.55	0.53	0.50
Non-Censored Data												
RMSE	NCGP	5.85	5.88	5.99	6.16	6.39	6.65	6.96	7.31	7.70	8.12	8.56
	NCGP-A	6.72	6.72	6.72	6.72	6.72	6.72	6.72	6.72	6.72	6.72	6.72
	CGP	6.21	6.11	6.01	6.01	6.05	6.09	6.13	6.28	6.42	6.51	6.68
R^2	NCGP	0.71	0.71	0.70	0.68	0.66	0.63	0.59	0.55	0.50	0.45	0.38
	NCGP-A	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62
	CGP	0.68	0.69	0.70	0.70	0.69	0.69	0.68	0.67	0.65	0.64	0.62

Table 3.5: Model Performance for Super-hub 2

Censorship Intensity:		0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Entire Dataset												
RMSE	NCGP	7.27	7.32	7.45	7.62	7.87	8.19	8.56	8.99	9.46	9.96	10.5
	NCGP-A	7.35	7.35	7.35	7.35	7.35	7.35	7.35	7.35	7.35	7.35	7.35
	CGP	7.42	7.26	7.20	7.17	7.19	7.25	7.29	7.37	7.42	7.50	7.65
R^2	NCGP	0.54	0.53	0.51	0.49	0.46	0.41	0.36	0.29	0.22	0.13	0.03
	NCGP-A	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53
	CGP	0.52	0.54	0.55	0.55	0.55	0.54	0.53	0.52	0.52	0.51	0.49
Non-Censored Data												
RMSE	NCGP	6.99	7.02	7.10	7.20	7.36	7.58	7.85	8.16	8.50	8.88	9.28
	NCGP-A	7.02	7.02	7.02	7.02	7.02	7.02	7.02	7.02	7.02	7.02	7.02
	CGP	7.12	7.01	6.97	6.92	6.96	6.98	6.99	7.05	7.04	7.09	7.22
R^2	NCGP	0.55	0.55	0.54	0.53	0.50	0.47	0.44	0.39	0.34	0.28	0.21
	NCGP-A	0.55	0.55	0.55	0.55	0.55	0.55	0.55	0.55	0.55	0.55	0.55
	CGP	0.54	0.55	0.56	0.56	0.56	0.55	0.55	0.54	0.55	0.54	0.52

Table 3.6: Model Performance for Super-hub 3

γ	c	Entire Dataset						Only Non-Censored					
		RMSE			R^2			RMSE			R^2		
		NCGP	NCGP-A	CGP	NCGP	NCGP-A	CGP	NCGP	NCGP-A	CGP	NCGP	NCGP-A	CGP
.1	0	31.295	32.867	30.261	.482	.429	.516	30.319	32.036	29.405	.512	.455	.541
	.1	31.717	32.867	30.216	.468	.429	.517	30.721	32.036	29.360	.499	.455	.543
	.2	31.964	32.867	30.112	.460	.429	.521	31.002	32.036	29.257	.490	.455	.546
	.3	32.174	32.867	30.089	.453	.429	.521	31.228	32.036	29.225	.482	.455	.547
	.4	31.975	32.867	30.174	.460	.429	.519	30.963	32.036	29.316	.491	.455	.544
	.5	31.982	32.867	30.156	.459	.429	.519	30.945	32.036	29.278	.492	.455	.545
	.6	32.180	32.867	30.217	.453	.429	.517	31.108	32.036	29.323	.486	.455	.544
	.7	32.308	32.867	30.167	.448	.429	.519	31.204	32.036	29.279	.483	.455	.545
	.8	32.525	32.867	30.063	.441	.429	.522	31.385	32.036	29.155	.477	.455	.549
	.9	32.898	32.867	29.976	.428	.429	.525	31.714	32.036	29.071	.466	.455	.551
	1	33.487	32.867	29.849	.407	.429	.529	32.256	32.036	28.935	.448	.455	.556
.2	0	31.294	32.555	30.601	.482	.440	.505	29.446	30.838	28.803	.539	.495	.559
	.1	31.862	32.555	30.050	.463	.440	.523	29.889	30.838	28.258	.525	.495	.576
	.2	32.347	32.555	29.928	.447	.440	.527	30.460	30.838	28.155	.507	.495	.579
	.3	32.668	32.555	29.855	.436	.440	.529	30.654	30.838	28.068	.501	.495	.581
	.4	32.845	32.555	30.019	.430	.440	.524	30.651	30.838	28.166	.501	.495	.578
	.5	33.060	32.555	30.156	.422	.440	.519	30.753	30.838	28.276	.497	.495	.575
	.6	33.509	32.555	30.284	.406	.440	.515	31.101	30.838	28.365	.486	.495	.572
	.7	34.394	32.555	30.337	.375	.440	.514	31.882	30.838	28.390	.460	.495	.572
	.8	35.526	32.555	30.350	.333	.440	.513	32.887	30.838	28.377	.425	.495	.572
	.9	36.736	32.555	30.178	.287	.440	.519	33.974	30.838	28.235	.387	.495	.576
	1	38.077	32.555	29.998	.234	.440	.524	35.175	30.838	28.038	.343	.495	.582
.3	0	31.294	32.444	31.032	.482	.444	.491	28.489	29.673	28.259	.563	.525	.570
	.1	31.864	32.444	30.310	.463	.444	.514	28.820	29.673	27.532	.552	.525	.591
	.2	32.360	32.444	29.887	.446	.444	.528	29.311	29.673	27.087	.537	.525	.604
	.3	32.793	32.444	29.843	.432	.444	.529	29.571	29.673	27.041	.529	.525	.606
	.4	33.019	32.444	29.917	.424	.444	.527	29.476	29.673	27.051	.532	.525	.606
	.5	33.956	32.444	30.291	.391	.444	.515	30.220	29.673	27.318	.508	.525	.598
	.6	35.344	32.444	30.594	.340	.444	.505	31.390	29.673	27.582	.469	.525	.590
	.7	36.898	32.444	30.800	.280	.444	.499	32.719	29.673	27.775	.423	.525	.584
	.8	38.704	32.444	30.719	.208	.444	.501	34.206	29.673	27.685	.369	.525	.587
	.9	40.624	32.444	30.572	.128	.444	.506	35.814	29.673	27.466	.309	.525	.593
	1	42.594	32.444	30.492	.041	.444	.509	37.487	29.673	27.453	.242	.525	.594
.4	0	31.297	32.914	31.607	.482	.427	.472	27.395	29.079	27.873	.582	.529	.567
	.1	32.070	32.914	30.679	.456	.427	.502	27.824	29.079	26.956	.568	.529	.595
	.2	32.770	32.914	30.054	.432	.427	.523	28.541	29.079	26.287	.546	.529	.615
	.3	33.279	32.914	29.844	.415	.427	.529	28.420	29.079	25.904	.550	.529	.626
	.4	34.145	32.914	29.946	.384	.427	.526	28.868	29.079	25.886	.536	.529	.627
	.5	35.369	32.914	30.274	.339	.427	.516	29.726	29.079	26.063	.507	.529	.621
	.6	37.184	32.914	30.547	.269	.427	.507	31.149	29.079	26.200	.459	.529	.617
	.7	39.396	32.914	30.973	.180	.427	.493	32.926	29.079	26.510	.396	.529	.608
	.8	41.949	32.914	30.973	.070	.427	.493	34.950	29.079	26.583	.319	.529	.606
	.9	44.679	32.914	30.929	-.055	.427	.494	37.051	29.079	26.627	.235	.529	.605
	1	47.582	32.914	30.953	-0.197	.427	.494	39.378	29.079	26.664	.136	.529	.604

Table 3.7: Model Performance in RandDropoff Experiments

Modeling Latent Mobility Demand via Quantile Regression Neural Networks

As explained in Chapter 3, observations of mobility demand are often “right-censored”, as they are obtained through usage of shared mobility services and thus inherently limited by available supply. Chapter 3 uses Gaussian Processes to flexibly learn the latent, non-censored demand distribution.

Motivated by some limits of Gaussian Processes, this Chapter suggests another non-parametric method to fit the latent distribution, based on Censored Quantile Regression Neural Networks (CQRNN). Previous works have shown some advantages of CQRNN in censored regression, yet apparently no works have applied CQRNN models in the Transportation domain. We apply CQRNN to both synthetic datasets and two real-world datasets from shared mobility providers in the Copenhagen metropolitan area. The results show that CQRNN can estimate the intended distributions better than both censorship-unaware models and parametric censored models.

I am the main author of this joint work with Assoc. Prof. Filipe Rodrigues and my main supervisor, Prof. Francisco C. Pereira. Within the theme of my thesis, this work offers an alternative non-parametric method for estimating the full uncertainty structure of latent demand from censored observations. We shall next submit this work to the 10th symposium of the European Association for Research in Transportation (hEART2021), and further extend it to a journal submission through additional case studies and comparison to other CQRNN implementations.

4.1 Introduction

Shared mobility services – e.g., taxis, bike-sharing and ridesourcing – offer several socio-economic benefits, such as reduced emissions, less traffic congestion and less need for parking (Santos, 2018, Smith and Hensher, 2020). Effective planning and deployment of such services require reliable estimates of mobility demand, as also obtainable from data-driven modeling (Laporte et al., 2018, Profillidis and Botzoris, 2019).

The data used in demand modeling is often derived from observations of service

usage, which are thus inherently limited by available vehicle supply. Moreover, the data of any mobility service provider does not account for demand lost to competing services and other transport modes. Consequently, actual demand for mobility is typically latent (i.e., unknown) and its observations are likely to lie below it, namely, they are often *right-censored*.

A joint work with Gammelli et al. (2020) shows that censorship-aware models can better estimate latent mobility demand, thereby allowing service providers to make better informed decisions. The models in that work are based on censored Gaussian Processes, which yield a full distribution of latent demand. However, while Gaussian Processes allow for a flexible, non-parametric fit, they still impose a Gaussian assumption on the latent distribution, and they also face some limitations when scaling to big datasets.

In this work, we propose to model latent mobility demand via Censored Quantile Regression Neural Networks (CQRNN), which are also non-parametric, yet do not face the aforementioned limitations. CQRNN estimates quantiles of the latent distribution via Neural Networks while accounting for censorship in the observed demand. We empirically demonstrate the advantages of CQRNN on synthetic datasets and apply CQRNN to demand estimation in real-world shared mobility data.

To the best of our knowledge, this is the first work to apply Censored Quantile Regression Neural Networks in the Transport domain. Furthermore, whereas existing works on CQRNN often assume fixed censorship, we conduct experiments with dynamic, random censorship. We also provide the Python implementation of our CQRNN models in <https://github.com/inon-peled/cqrnn-pub/>.

The rest of this work is organized as follows. In Section 4.4.1, we review works related to CQRNN and identify knowledge gaps, particularly in the Transport domain. Section 4.3 then describes our CQRNN methodology, and we demonstrate its advantages via experiments on synthetic data in Section 4.4. In Section 4.5, we apply our methodology to censored datasets from two shared mobility services in Denmark: bike-sharing and shared Electric Vehicles. Finally, Section 4.6 summarizes our findings and outlines our future work plans.

4.2 Literature Review

In this Section, we review existing works on non-censored and censored Quantile Regression (QR), and in particular, Neural Network-based QR. For the more general topics of Censored Regression and Neural Networks, we kindly refer the reader to the following resources. A review of censored and non-censored

methods for mobility demand modeling appears in our recent joint work with Gammelli et al. (2020). The general theory and practice of Neural Networks is well studied in (Nielsen, 2015), and several of their recent applications in the transport domain are reviewed in (Himanen et al., 2019).

Let us first introduce the general method of Quantile Regression (Koenker and Bassett, 1978), regardless of censorship. For any probability distribution and $0 < \theta < 1$, the θ 'th *quantile* is the smallest value at which the cumulative probability mass is θ . QR approximates a latent distribution by estimating its quantiles for given θ 's, and thus does not presume any parametric form for the distribution. The regression itself can follow any functional form – e.g., linear (Koenker and Bassett, 1978), nonlinear (Koenker and Park, 1996), multivariate (Carlier et al., 2016) or nonparametric (Zheng, 2012) – and the quantiles can be combined into a fully estimated distribution (Cannon, 2011, Quiñonero-Candela et al., 2006).

Importantly, the fully estimated distribution preserves useful information, which might otherwise be lost through the more common practice of estimating only a few central moments, such as mean and standard deviation (Peled et al., 2019a). In turn, the preserved information allows for better informed decisions, e.g., service operators can use the full uncertainty structure of future demand to decide whether to balance the fleet conservatively or more opportunistically. In addition, by taking values of θ close to 0 and 1, QR can be more robust to outliers than mean regression (Koenker, 2005).

A variant of QR is Censored Quantile Regression (CQR), where observed realizations from a latent distribution are assumed to be clipped at some thresholds. The distribution of these observations is then a mixture of continuous (within the observable range) and discrete (at each threshold). Many works on CQR build upon the early formulation by Powell (1984, 1986), as we ultimately do too in Section 4.3. Some of these works focus on estimators of derived CQR formulations (Buchinsky, 1998, Cheng and Tzeng, 2014, Chernozhukov et al., 2015, Galvao et al., 2013, Leng and Tong, 2013). Others works on CQR develop more complex and non-parametric models, as we next review, starting with models that are not based on Neural Networks.

Yu and Stander (2007) devise a Bayesian Inference approach to linear CQR, in which they use the Asymmetric Laplace likelihood, as also common in other Bayesian CQR works (Yang et al., 2016a). They evaluate their approach on some commonly used, censored data baselines: synthetic datasets, which we too use in Section 4.4, and a real-world dataset of women's labor force participation (Mroz, 1987). Gannoun et al. (2005) develop a CQR model based on local linear approximations and evaluate it on synthetically generated datasets. Shim and Hwang (2009) offer a Support Vector Machine-based CQR model, which they

evaluate on the same synthetic datasets as in Gannoun et al. (2005), and on a real-world dataset of heart transplant survival. Li and Bradic (2020) propose a Random Forest-based CQR model, which they compare with other Random Forest models on both synthetic data and two real-world datasets: housing prices and a biliary disease.

In the last two decades, Neural Networks have increasingly been used for Quantile Regression (QRNN) in multiple research areas, taking advantage of their flexible, nonlinear modeling capabilities. For non-censored regression, Taylor (2000) provides an early form of QRNN with a single dense hidden layer, and uses it to estimate a latent distribution of multiperiod financial returns. In studies of the electric power industry, He et al. use non-censored QRNN to estimate latent distributions of electricity production (He and Li, 2018, He and Zhang, 2020) and consumption (He et al., 2019), while Haben and Giasemidis (2016) and Zhang et al. (2018) use non-censored QRNN to predict electricity loads. In the Transport domain, (Tian et al., 2020) use a non-censored QRNN with a single hidden neuron to predict 15 min air traffic in a Chinese airport, and Rodrigues and Pereira (2020) devise a non-censored, multi-output QRNN that jointly estimates mean and quantiles, whereby they predict 30 min taxi demand in New York City.

Very few works apply QRNN in a Censored setting (CQRNN). Cannon (2011) develops a general architecture for both censored and non-censored QRNN, which he implements as an R package. He uses a smoothing technique by Chen (2007) to replace the loss function with a differentiable approximation, which is amenable to gradient-based training, and applies the implementation in a censored case study of precipitation forecasting. Jia and Jeong (2020) propose another CQRNN model, similar to but deeper than that of (Cannon, 2011). They implement their model in Python via *Keras* and apply it to censored survival datasets: a synthetic dataset and a breast cancer dataset.

Except for (Cannon, 2011) and (Jia and Jeong, 2020), we are not aware of other works that use Quantile Regression Neural Networks in Censored case studies. In particular, there appear to be no works on CQRNN in the transport domain, despite the prevalent censorship in transport data (Section 4.1), the flexibility of Neural Networks and the advantages of Quantile Regression. We next address this gap by applying CQRNN to several datasets of real-world shared mobility services.

Figure 4.1: Our Censored Quantile Regression Neural Network architecture.

4.3 Methodology

A censored dataset consists of covariates $\mathbf{x}_1, \dots, \mathbf{x}_N$ and corresponding observations $y_1, \dots, y_N$, which are clipped versions of latent variables $y_1^*, \dots, y_N^*$. Namely, for some thresholds $\tau_1, \dots, \tau_N$, all the observations are either left-censored or right-censored, such that:

$$y_i = \begin{cases} y_i^*, & y_i^* > \tau_i \\ \tau_i, & y_i^* \leq \tau_i \end{cases} \text{ in left-censorship,} \quad (4.1)$$

$$y_i = \begin{cases} y_i^*, & y_i^* < \tau_i \\ \tau_i, & y_i^* \geq \tau_i \end{cases} \text{ in right-censorship.} \quad (4.2)$$

Each threshold is either given or unknown, and if censorship is fixed, then $\tau_1 = \dots = \tau_N$. $y_1^*, \dots, y_N^*$ are drawn from a latent distribution, whose θ 'th quantiles we wish to estimate for some $0 < \theta < 1$. To this end, we construct, fit and evaluate various Neural Networks (NNs), as follows.

We construct NNs independently for each θ 'th quantile, as illustrated in Figure 4.1. All the NNs are Feed-Forward and optionally use input Dropout (Srivastava et al., 2014) for regularization. Throughout the experiments, we vary the Hidden layers and the activation function of the Aggregate Outputs layer. To experiment with right-censored datasets, we also negate each NN input and output as well as mirror the output quantiles, e.g., swap the output for $\theta = 0.05$ with the output for $\theta = 0.95$.

Fitting is done by minimizing a dedicated loss function, which we next describe. As mentioned in Section 4.2, Yu and Stander (2007) use the following likelihood

Figure 4.2: Tilted Loss.

function for Bayesian Inference with left-censorship at zero:

$$\mathcal{S} \left(y_1^{(j)}, \dots, y_N^{(j)} \middle| \beta^{(j)}, \mathbf{x}_i, \theta \right) = \theta^N (1 - \theta)^N \exp \left\{ - \sum_{i=1}^N \rho_\theta \left(y_i - \max \left\{ 0, \mathbf{x}_i^\top \beta^{(j)} \right\} \right) \right\}, \quad (4.3)$$

where $\rho : \mathbb{R} \rightarrow \mathbb{R}$ is the Tilted Loss (TL) function,

$$\rho_\theta(r) = \max\{\theta r, (\theta - 1)r\}. \quad (4.4)$$

Figure 4.2 illustrates how TL penalizes the prediction error $r = \hat{q}_\theta - y$ in a manner that depends on θ . For the median ($\theta = 0.5$), the loss is the same regardless of the sign of r . For quantiles above the median (e.g., $\theta = 0.95$), the loss is worse for $y > \hat{q}_\theta$ than for $y < \hat{q}_\theta$ with the same magnitude of r , and vice versa for quantiles below the median (e.g., $\theta = 0.05$). For any θ , the loss equals zero if $y = \hat{q}_\theta$ and is otherwise positive.

Based on (4.3), we use the following likelihood function:

$$\mathcal{C} \left(y_1^{(j)}, \dots, y_N^{(j)} \middle| \beta^{(j)}, \mathbf{x}_i, \theta \right) = \theta^N (1 - \theta)^N \exp \left\{ - \sum_{i=1}^N \rho_\theta \left(y_i - \max \left\{ \tau_i, \hat{q}_{\theta,i} \right\} \right) \right\}, \quad (4.5)$$

where $\hat{q}_{\theta,i}$ is the NN output; note that τ_i must be specified also for non-censored observations. The Negative Log-Likelihood (NLL) of (4.5) is the loss function for fitting.

We fit each NN using backpropagation and the Adam optimizer (Kingma and Ba, 2014) with norm clipping at 1, while using a validation set for early stop. For hyper-parameter optimization, we experiment with Adam learning rates in $\{0.001, 0.01, 0.1, 1\}$, and choose the one that performs best on the validation set. Additional details about architecture, initialization, fitting and evaluation depend on the dataset and manner of experimentation, as in the following Sections.

4.4 Experiments for Demonstrating the Advantages of Censored Quantile Regression

In this Section, we empirically demonstrate some advantages of Censored Quantile Regression Neural Networks (CQRNN). First, we compare CQRNN with censorship-unaware Quantile Regression and show that CQRNN can better reconstruct latent values. Then, we compare CQRNN to parametric Censored Regression and show an advantage and disadvantage of each modeling method. The experiments in this Section are based on commonly used, synthetic baseline datasets, and the next Section proceeds to use real-world transport datasets.

4.4.1 Non-censored vs. Censored Quantile Regression

Let us first show that the predictive quality of Quantile Regression Neural Networks can improve by accounting for data censorship. For this, we use some common synthetic baseline datasets, as in Yu and Stander (2007), where the latent variable is

$$y^* = x_0 + x_1 + x_2 + \varepsilon, \quad (4.6)$$

where $x_0 = 1, x_1 \in \{-1, 1\}, x_2 \in \mathbb{R}$ and the noise ε follows some distribution with 0 mean. Left-censorship occurs at zero, so that we observe

$$y = \max\{0, y^*\}. \quad (4.7)$$

For any random variable A and $0 < \theta < 1$, let $q_\theta(A|\mathbf{x})$ denote the θ 'th conditional quantile of A given $\mathbf{x} = [x_0, x_1, x_2]^\top$. Hence:

$$q_\theta(y|\mathbf{x}) = \max\{0, q_\theta(y^*|\mathbf{x})\} = \max\{0, x_0 + x_1 + x_2 + q_\theta(\varepsilon|\mathbf{x})\}. \quad (4.8)$$

Similarly to (Yu and Stander, 2007), we experiment with $\theta = 0.05, 0.50, 0.95$ and three noise distributions,

$$\text{Standard Gaussian: } \varepsilon^{(1)} \sim \mathcal{N}(0, 1), \quad (4.9)$$

$$\text{Heteroskedastic: } \varepsilon^{(2)} \sim (1 + x_2)\mathcal{N}(0, 1), \quad (4.10)$$

$$\text{Gaussian Mixture: } \varepsilon^{(3)} \sim 0.75\mathcal{N}(0, 1) + 0.25\mathcal{N}(0, 2^2). \quad (4.11)$$

The corresponding conditional quantiles of y are thus

$$q_\theta(y^{(j)}|\mathbf{x}) = \max\{0, q_\theta(y^{*,j}|\mathbf{x})\}, \quad (4.12)$$

Figure 4.3: Distribution of synthetic y^* for $x_0 = 1, x_1 = -1, x_2 = -0.5, 0.5, 1.5$ and each noise distribution $\varepsilon^{(j)}$. In each case, y^* follows a Gaussian distribution with mean $x_0 + x_1 + x_2$ (dash-dotted line). For the Standard Gaussian noise (dashed) and Gaussian Mixture noise (solid), the distribution of y^* is homoskedastic, i.e., has fixed variance 1 and 0.791^2 , respectively. For the Heteroskedastic noise (dotted), y^* has variance that changes with x_2 as $(1 + x_2)^2$.

Figure 4.4: Conditional quantiles of synthetic y^* for each noise distribution.

such that:

$$q_\theta(y^{*,1}|\mathbf{x}) = \Phi^{-1}\left(x_0 + x_1 + x_2, 1\right), \quad (4.13)$$

$$q_\theta(y^{*,2}|\mathbf{x}) = \Phi^{-1}\left(x_0 + x_1 + x_2, |1 + x_2|\right), \quad (4.14)$$

$$q_\theta(y^{*,3}|\mathbf{x}) = \Phi^{-1}\left(x_0 + x_1 + x_2, \sqrt{0.75^2 + 0.25^2}\right), \quad (4.15)$$

where $\Phi^{-1}(\mu, \sigma) : [0, 1] \rightarrow \mathbb{R}$ is the quantile function of $\mathcal{N}(\mu, \sigma^2)$. Figure 4.3 illustrates the distribution of y^* with each noise, for $x_1 = 1$ and several values of x_2 . Figure 4.4 illustrates the conditional quantiles of y^* for each noise and θ . For Heteroskedastic noise, the conditional quantiles of y^* are non-linear, as their slopes change at $x_2 = -1$.

For each $\varepsilon^{(j)}$, $j = 1, 2, 3$, we generate a synthetic dataset by independently

Dataset	$\theta = 0.05$	$\theta = 0.50$	$\theta = 0.95$
Standard Gaussian	62.7%	23.9%	2.0%
Heteroskedastic	68.0%	23.9%	11.4%
Gaussian Mixture	54.1%	23.9%	4.6%

Table 4.1: Percent of zero conditional quantiles for the censored observations in each synthetically generated dataset.

drawing $N = 1000$ samples from $\varepsilon^{(j)}$ and $\mathbf{x}$, where

$$x_0 = 1, \quad x_1 \sim \mathcal{U}\{-1, 1\}, \quad x_2 \sim \mathcal{N}(0, 1). \quad (4.16)$$

We then also compute the corresponding y^* and y , and obtain that approximately 30% of the observations $y_1^{(j)}, \dots, y_N^{(j)}$ are censored. Further, for each θ , Table 4.1 provides the percent of zeros among the conditional quantiles $q_\theta(y_1^{(j)}|\mathbf{x}_1^{(j)}), \dots, q_\theta(y_N^{(j)}|\mathbf{x}_N^{(j)})$. The most challenging cases to model are those with $\theta = 0.05$, where the conditional quantiles are particularly prone to censorship.

For each $j = 1, 2, 3$, we fix train, test and validation sets by randomly partitioning the j 'th dataset as 62% : 15% : 33%, respectively. To model the θ 'th quantile, we use several Neural Networks (NNs), each consisting of a single neuron with activation function η . Namely, each NN takes as input $\mathbf{x}$ and outputs

$$\hat{q}_\theta(y|\mathbf{x}) = \eta(\mathbf{x}^\top \boldsymbol{\beta}), \quad (4.17)$$

where $\boldsymbol{\beta}$ are trainable weights. We fit each NN on the train set via backpropagation, using Adam optimization with learning rate 0.01 and norm clipping at 1. Weights are initialized to 1, and in each training epoch, the whole train set is processed in a single batch. Training stops when the validation loss does not improve for 10 consecutive epochs.

First, as an example of a common model that ignores censorship, we use a non-censored linear model, where η is the identity function and the loss to be minimized is

$$\rho_\theta(\hat{q}_\theta(y|\mathbf{x}) - y). \quad (4.18)$$

Next, we turn to censorship-aware models, the first of which is $\mathcal{C}+\Sigma$, where η is the identity function, so that the model is still linear as in (Yu and Stander, 2007). The second censorship-aware model is non-linear, such that η is the Exponential Linear Unit,

$$\text{ELU}(z) = \begin{cases} z, & z > 0 \\ \exp(z) - 1, & z \leq 0 \end{cases}. \quad (4.19)$$

Finally, we measure the predictive performance of each NN on the test set against the actual conditional quantiles of y^* in Equations (4.13) to (4.15). The measures we use are Coefficient of Determination (R^2), Mean Absolute Error (MAE) and Rooted Mean Squared Error (RMSE). For any dataset, these measures are defined as follows, omitting superscripts and function arguments for better legibility:

$$R^2 = 1 - \frac{\sum_{i=1}^N (\hat{q}_{\theta,i} - q_{\theta,i})^2}{\sum_{i=1}^N (\hat{q}_{\theta,i} - \bar{q}_{\theta})^2}, \quad (4.20)$$

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |\hat{q}_{\theta,i} - q_{\theta,i}|, \quad (4.21)$$

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^N (\hat{q}_{\theta,i} - q_{\theta,i})^2}, \quad (4.22)$$

where $\bar{q}_{\theta}$ is the mean of $q_{\theta,1}, \dots, q_{\theta,N}$. Better predictive quality corresponds to R^2 closer to 1 and MAE and RMSE closer to 0.

The results appear in Table 4.2 on the following page, which shows that the CQRNN models outperform the censorship-unaware model. This holds when evaluating on either the entire test set or its non-censored subset – where the latent values are revealed. For the median ($\theta = 0.5$), the linear CQRNN is mostly the best model, whereas for the more challenging $\theta = 0.05, 0.95$, the non-linear CQRNN is mostly the best model.

4.4.2 Parametric vs. Non-Parametric Censored Quantile Regression

Since its introduction by Tobin (1958), the Tobit model has become a cornerstone of parametric censored modeling. Tobit assumes that the latent variable depends on covariates linearly with Gaussian white noise, and is censored at a given fixed threshold. Hence in Tobit, the latent quantiles for the i 'th observation are given by the parametric distribution

$$\mathcal{N}(\mathbf{x}_i^T \boldsymbol{\beta}, \sigma^2), \quad (4.23)$$

where $\mathbf{x}_i$ are covariates, $\boldsymbol{\beta}$ are linear coefficients to be estimated, and σ is standard deviation, either given or to be estimated too. Further, the Tobit likelihood for upper censorship is

$$\prod_{i=1}^N \left\{ \frac{1}{\sigma} \varphi \left(\frac{y_i - \mathbf{x}_i^T \boldsymbol{\beta}}{\sigma} \right) \right\}^{1-l_i} \left\{ 1 - \Phi \left(\frac{y_i - \mathbf{x}_i^T \boldsymbol{\beta}}{\sigma} \right) \right\}^{l_i}, \quad (4.24)$$

		Standard Gaussian			Heteroskedastic			Gaussian Mixture		
θ	Model	R^2	MAE	RMSE	R^2	MAE	RMSE	R^2	MAE	RMSE
All Test Data										
0.05	TL+ Σ	0.220	0.983	1.230	-0.500	1.205	1.558	0.528	0.766	0.957
	C+ Σ	0.499	0.793	0.986	-0.418	1.179	1.515	0.715	0.601	0.744
	C+ELU	0.690	0.594	0.775	-0.376	1.159	1.493	0.744	0.555	0.705
0.50	TL+ Σ	0.904	0.346	0.433	0.904	0.351	0.432	0.909	0.336	0.421
	C+ Σ	1.000	0.015	0.018	1.000	0.015	0.018	1.000	0.014	0.017
	C+ELU	0.990	0.069	0.141	0.987	0.096	0.156	0.989	0.090	0.149
0.95	TL+ Σ	0.979	0.156	0.200	0.811	0.855	1.035	0.974	0.176	0.222
	C+ Σ	0.985	0.137	0.172	0.913	0.412	0.703	0.983	0.153	0.183
	C+ELU	0.973	0.184	0.230	0.947	0.374	0.547	0.982	0.155	0.185
Only Non-Censored										
0.05	TL+ Σ	0.135	0.443	0.591	-0.420	0.275	0.336	0.422	0.402	0.535
	C+ Σ	0.641	0.295	0.381	-0.441	0.279	0.339	0.585	0.347	0.453
	C+ELU	0.742	0.246	0.323	-0.414	0.276	0.336	0.602	0.340	0.444
0.50	TL+ Σ	0.924	0.237	0.279	0.914	0.250	0.297	0.928	0.230	0.272
	C+ Σ	1.000	0.016	0.019	1.000	0.016	0.019	1.000	0.015	0.018
	C+ELU	0.998	0.036	0.046	0.993	0.069	0.088	0.994	0.066	0.077
0.95	TL+ Σ	0.980	0.147	0.185	0.798	0.777	0.947	0.976	0.158	0.192
	C+ Σ	0.985	0.130	0.160	0.891	0.387	0.694	0.979	0.153	0.183
	C+ELU	0.972	0.177	0.218	0.933	0.360	0.545	0.979	0.152	0.182

Table 4.2: Predictive quality for conditional quantiles of y^* in the synthetic datasets.

where φ is the Probability Density Function (PDF) of $\mathcal{N}(0, 1)$, Φ is its Cumulative Distribution Function (CDF), and for a given fixed threshold τ :

$$l_i = \begin{cases} 0, & y_i < \tau \\ 1, & y_i = \tau \end{cases}. \quad (4.25)$$

Let us now compare Tobit parametric modeling to non-parametric CQRNN, using the same synthetic baseline datasets as above. For both modeling methods, we use an NN with a single linear neuron, which we fit similarly to Section 4.4.1. When fitting Tobit, we fix $\sigma = 1$ and use the NLL of (4.24) as the loss function, whereas when fitting CQR for $\theta = 0.05, 0.95$, we use the NLL of (4.5) as the loss function.

Finally, we evaluate the performance via two common measures of Quantile Regression:

$$\text{ICP} = \text{fraction of } y^* \text{ within the estimated } 5\% - 95\% \text{ interval.} \quad (4.26)$$

$$\text{MIL} = \text{mean length of the estimated } 5\% - 95\% \text{ interval.} \quad (4.27)$$

ICP should be close to $0.95 - 0.05 = 0.9$, while MIL should converge to the

Dataset	Model	All Test Data		Only Non-Censored	
		ICP	MIL	ICP	MIL
Standard Gaussian	Tobit	0.909	3.290	0.915	3.290
	$\mathcal{C} + \Sigma$	0.888	3.155	0.885	3.235
Heteroskedastic	Tobit	0.936	3.290	0.941	3.290
	$\mathcal{C} + \Sigma$	0.779	3.742	0.907	4.480
Gaussian Mixture	Tobit	0.852	2.601	0.852	2.601
	$\mathcal{C} + \Sigma$	0.721	2.644	0.907	3.081

Table 4.3: Non-parametric QR vs. parametric Tobit on synthetic datasets. Best ICP is highlighted in bold, and ties are broken by MIL.

latent MIL. Among models with same ICP, we thus prefer the one that yields the lowest MIL.

Table 4.3 summarizes the performance of Tobit and CQR. As expected, Tobit performs best on the synthetic dataset with Standard Gaussian noise, which most closely matches its modeling assumptions. When evaluated on all test observations, Tobit outperforms QR by obtaining ICP closer to the desired 0.9, for each synthetic dataset. However, when evaluated on just the non-censored test observations (approx. 30% of each dataset), where the true values are reliably known, QR outperforms Tobit while also maintaining ICP close to 0.9. The results thus suggest that CQRNN tends to yield flatter distributions (higher MIL) that better approximate the latent distribution of non-censored observations.

4.5 Experiments for Estimating Latent Mobility Demand

In this Section, we apply Censored Quantile Regression Neural Network (i.e., CQRNN) to real-world data from shared mobility services. Contrary to the synthetic datasets in the previous Section, real-world datasets do not feature the latent variable. Hence similarly to (Gammelli et al., 2020), we treat the available data as y^* and manually censor it per various censorship schemes. We then fit several CQRNN models for $\theta = 0.05, 0.95$ and evaluate them via ICP (4.26) and MIL (4.27).

First, we use a censorship-unaware NN with a single linear unit (denoted as Σ), which we train to minimize plain Tilted Loss (4.18). Then, we equip the same architecture with censorship-awareness, using (4.5) as loss (denoted as $\mathcal{C}$). Thereafter, we experiment with non-linear activations, and finally add a Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) in the

Figure 4.5: Bike-sharing hubs and aggregated superhubs, as in (Gammelli et al., 2020): 1) Nørrebro, 2) Kgs. Nytorv, 3) København H.

Hidden layer, immediately before a linear Output Aggregation layer (Figure 4.1).

We also experiment with regularization (denoted as “Reg”) by randomly eliminating 20% of the input through the Dropout layer and using ℓ_2 regularization in all subsequent layers. We have also experimented with either 1 or 10 stacked units with non-linear activations (tanh, sigmoid, ELU, RELU (Nielsen, 2015)) in the Hidden layer, either with or without regularization. However, these additional models performed poorly, as we further discuss in Section 4.6, and so we omit their results.

4.5.1 Bike-sharing Data

The first real-world dataset is obtained from Donkey Republic, a bike-sharing service provider in the Copenhagen metropolitan area in Denmark. The data consists of pickups and returns of bicycles in predefined hubs, which we aggregate spatially by “superhubs” and temporally by no. daily pickups daily, as in (Gammelli et al., 2020). superhubs rarely run out of bicycles at any moment, hence this data represents actual demand quite well.

We censor the data as follows:

1. Randomly select a γ portion of all y_i^* .
2. For each selected y_i^* , independently sample

$$\delta_i \sim \mathcal{U}[c_1, c_2], \quad (4.28)$$

and let

$$y_i = (1 - \delta_i)y_i^*. \quad (4.29)$$

Figure 4.6: Mean $|ICP - 0.9|$ (left) and MIL (right) for bike-sharing data, evaluated on all test observations. Lower is better.

Our experiments use $\gamma = 0.0, 0.1, \dots, 0.9$ and $(c_1, c_2) = (0.01, 0.33), (0.34, 0.66), (0.67, 0.99)$. For each γ, c_1, c_2 , we independently censor the data 10 times to obtain differently censored datasets $B_1, \dots, B_{10}$, and we partition each B_j consecutively into train, validation and test sets with equal proportions. We then fit each NN model independently for 10 random initializations of weights, drawn independently from $\mathcal{N}(0, 1)$. The covariates are the 7 previous observation lags, and the censorship thresholds for non-censored observations are

$$\tau_i^{(j)} = y_i^{(j)} \times \frac{\text{mean of train } y^*}{\text{mean of train } y^{(j)}}. \quad (4.30)$$

After fitting, we evaluate ICP and MIL for each γ, c_1, c_2 as follows. First, for each $B_1, \dots, B_{10}$, we consider only initializations that yield reasonable validation MIL, as:

$$\frac{\text{validation MIL}}{\text{mean of train } y^{(j)}} \leq 2. \quad (4.31)$$

We then select the initialization that yields validation ICP closest to 0.9. Finally, we average the test ICP and test MIL over the 10 selected initializations.

The results for the bike-sharing data experiments are summarized in Figure 4.6 for the entire test set and Figure 4.7 on the next page for only non-censored test observations. To reduce visual clutter, we limit the vertical axis and omit results for non-regularized Σ with $\mathcal{C}$ loss, as they are mostly worse than regularized Σ with $\mathcal{C}$. In each Figure, rows range over superhubs, columns range over (c_1, c_2) ,

Figure 4.7: Mean $|ICP - 0.9|$ (left) and MIL (right) for bike-sharing data, evaluated on non-censored test observations. Lower is better.

and each horizontal axis ranges over γ . For ICP, we plot $|ICP - 0.9|$, namely, its distance from ideal ICP.

We see that a more complex architecture results in higher MIL, but also generally better ICP. The censorship-unaware model often yields the worst ICP, though as also found in (Gammelli et al., 2020), its ICP is occasionally better than that of the censorship-aware models, when relatively few observations are censored ($\gamma \leq 0.2$). The overall better performance of the censorship-aware models holds not only for the entire test set, but also for its non-censored subset, where the true (latent) values are known.

We also see that among CQRNN models, the LSTM-based model often yields better ICP than the purely linear model. An exception to this is the case of superhub 3 with $(c_1, c_2) = (0.67, 0.99)$, where the linear CQRNN yields the best ICP on both test subsets, possibly because this more challenging case requires better fitting of LSTM weights.

4.5.2 Shared Electric Vehicles Data

The second real-world dataset comes from Share Now, a shared Electric Vehicles (EVs) service operator in the Copenhagen metropolitan area too. The EVs are picked up from and returned to reserved parking areas. This dataset, which we denote as $\mathcal{D}_{EV}$, consists of 2.6 million trips in years 2016-2019, where each trip

Figure 4.8: Shared EV data and $\alpha = 40\%$ fleet reduction.

record contains the endpoints, driver ID and vehicle ID.

We wish to experiment with complete censorship of daily demand of EV mobility. First, we let $y_1^*, \dots, y_N^*$ be the daily no. trip starts in $\mathcal{D}_{EV}$. For censorship, we use the following scheme:

1. Randomly select an α portion of all vehicles in $\mathcal{D}_{EV}$.
2. Let $\mathcal{D}'_{EV}$ be $\mathcal{D}_{EV}$ without any trips that involve the selected vehicles.
3. Let $y_1, \dots, y_N$ be the daily no. trip starts in $\mathcal{D}'_{EV}$.

Consequently, every y_i is censored, so that

$$y_i \approx (1 - \alpha)y_i^*, \quad (4.32)$$

as illustrated in Figure 4.8.

Because all observations are censored, there is no need to specify any threshold for non-censored observations. For each $\alpha = 10\%, 20\%, 30\%, 40\%$, we independently apply the censorship scheme 10 times. For each of the 10 censored datasets thus obtained, we partition into train:validation:test as 1 : 1 : 1 and fit each NN with random initialization of weights, drawn independently from $\mathcal{N}(0, 1)$. Finally, we evaluate each NN by averaging its test ICP and test MIL over the 10 experiments.

The results appear in Table 4.4 on the next page, where we see that the censorship-unaware model mostly yields both the worst ICP and worst MIL. Regularization again improves results for the censorship-aware Σ with $\mathcal{C}$ loss. Among the censorship-aware models, the LSTM model often yields the best ICP, and otherwise has ICP close to the ICP of the regularized linear model. For censorship-aware models too, better ICP is accompanied by higher MIL, as in Section 4.5.1. As expected in complete censorship, all models deteriorate rapidly as α increases and yield ICP far below 0.9 for $\alpha = 0.4$, where our experiments thus stop.

Model	$\alpha = 10\%$		$\alpha = 20\%$		$\alpha = 30\%$		$\alpha = 40\%$	
	ICP	MIL	ICP	MIL	ICP	MIL	ICP	MIL
TL Σ	0.870 $\pm$ 0.0	1680 $\pm$ 123	0.659 $\pm$ 0.0	1442 $\pm$ 75	0.341 $\pm$ 0.1	1339 $\pm$ 84	0.106 $\pm$ 0.0	1105 $\pm$ 62
C Σ	0.858 $\pm$ 0.0	1594 $\pm$ 59	0.673 $\pm$ 0.0	1419 $\pm$ 64	0.342 $\pm$ 0.1	1242 $\pm$ 74	0.127 $\pm$ 0.0	1084 $\pm$ 107
C Reg+ Σ	0.912 $\pm$ 0.0	2017 $\pm$ 68	0.858 $\pm$ 0.0	1848 $\pm$ 84	0.683 $\pm$ 0.1	1641 $\pm$ 109	0.282 $\pm$ 0.1	1406 $\pm$ 48
C LSTM+ Σ	0.904 $\pm$ 0.0	2210 $\pm$ 165	0.850 $\pm$ 0.1	2014 $\pm$ 262	0.779 $\pm$ 0.1	2329 $\pm$ 1114	0.343 $\pm$ 0.2	1595 $\pm$ 302

Table 4.4: Results of experiments with shared EV data, as average $\pm$ standard deviation. Best ICP is highlighted in bold.

4.6 Conclusion

In this work, we propose to estimate the full distribution of latent mobility demand via Censored Quantile Regression Neural Networks (CQRNN). First, we demonstrate the advantages of CQRNN using synthetic baseline datasets with various noise distributions, both homoskedastic and heteroskedastic. We obtain that CQRNN outperforms censorship-unaware QRNN on both the entire test set and its non-censored subset, where the true values are reliably known. We also compare CQRNN to the common Tobit model, which assumes Gaussian white noise, and obtain that CQRNN tends to yield flatter distributions that better approximate the latent uncertainty structure of non-censored observations.

Next, we apply CQRNN to real-world datasets from two shared mobility services – bike-sharing and shared Electric Vehicles (EVs) – which we randomly censor either partially or completely. For both datasets, more complex CQRNN architectures yield higher MIL and generally better ICP – where the use of Long Short-Term Memory (LSTM) often leads to best performance – whereas censorship-unaware QRNN often yields the worst ICP.

The experiments on synthetic and real-world dataset thus lead to similar conclusions about the effectiveness of CQRNN for Censored Regression. One difference, however, is that the use of nonlinear activations improves performance for the synthetic datasets, yet deteriorates performance for the real-world datasets. The cause for this is unlikely to be overfitting, as we use regularization techniques. Instead, a possible cause is the non-differentiability of our censored loss function, which may thus benefit from smoothing as in (Cannon, 2011).

For future work, we therefore plan to experiment with a smooth approximation of the loss function, after which we shall compare our models to the models of (Cannon, 2011) and (Jia and Jeong, 2020). We also plan to experiment with multivariate Quantile Regression, e.g., similarly to (Rodrigues and Pereira, 2020). Further, we plan to take advantage of possible spatio-temporal correlations in the datasets, e.g., using Convolutional Neural Networks as in (Chu et al., 2019).

Predictive Optimization Framework for Demand-Responsive Public Transport

This Chapter contains my first mutual work with colleagues in Nanyang Technological University: my co-supervisor, Assoc. Prof. Justin Dauwels, and peer PhD student, Kelvin Lee; additional co-authors are Asst. Prof. Yu Jiang and my main supervisor, Prof. Francisco C. Pereira. We propose an online predictive optimization framework for dynamic operation of shared transit services (Peled et al., 2019b).

On the travel demand side, the framework uses Quantile Regression to estimate the marginal distributions of Origin-Destination pairs. It then combines the marginals into a joint demand distribution via a Gaussian copula, which captures the structure of correlation between the marginals. On the supply side, the framework uses a linear programming formulation to optimize service routes and frequency per the joint demand distribution. We evaluate the framework through a case study of autonomous mobility in the DTU Lyngby campus, as part of a real-world project (LINC, 2020), and obtain that it can outperform conventional methods for route optimization, which do not leverage full predictive distributions.

This is our first effort at explicitly interfacing Transportation demand and supply. I am the main author, while Yu Jiang and Kelvin Lee formulated and implemented the copula and the linear program. Eventually though, the optimization results did not demonstrate sufficient benefits for using the entire framework. We thus decided that I present only the demand modeling portion of the work in the 2019 IEEE Intelligent Transportation Systems Conference (ITSC) (Peled et al., 2019a).

For me, this was the first use of Quantile Regression in modeling the full uncertainty structure of mobility demand. Through this work, I also became better familiar with Public Transport optimization, and in particular, the challenges in making it demand-responsive. The data itself is somewhat novel too, as the crowd movements in it have been aggregated through WiFi probing.

5.1 Introduction

Various institutions around the world are increasingly incorporating autonomous vehicle fleets into their on-campus mobility solutions (Abdullah, 2018, Alton, 2018, LINC, 2020). Whereas the itineraries of traditional bus services are often pre-fixed, autonomous vehicle fleets can have their itineraries dynamically adapted in real-time, e.g., per changing demand for mobility during a work day. In other words, autonomous mobility services are amenable to demand-responsive routing, if demand can be predicted ahead of time. Fortunately, institutional campuses are often finely meshed with WiFi access points, which can in turn be used as sensors of crowd presence (Meneses and Moreira, 2012, Sevt-suk, 2009). By using WiFi information to detect movements of wireless devices across campus, the current and future demand for mobility can be estimated. An opportunity thus emerges for real-time, demand-responsive autonomous mobility.

Existing literature in transport is rich with methods for demand prediction and route optimization. However, as we present in Section 5.2, these studies often fall short of fully treating both these aspects. First, existing studies often concentrate either only on demand prediction or supply optimization, leaving the other to external works. Second, these studies often insufficiently account for uncertainty in demand and supply, and use point estimates instead of more informative full distributions. In this study, however, we offer a framework that explicitly incorporates both demand prediction and supply optimization, while leveraging demand uncertainty to optimize demand-responsive transit services.

5.1.1 Overview of Our Solution Framework

The setting in which we apply our framework is that of a mobility service, for instance, a fleet of autonomous shuttles in a university or hospital. People move in the serviceable area between connected locations, such as buildings connected by roads. These movements are captured by a network of sensors, e.g., a network of WiFi hotspots probing people’s wireless devices. The different locations form a partition of the spatial dimension, and we further partition the temporal dimension into consecutive lags (e.g., $1h$).

Crowd movements between different locations at different times are subject to *uncertainty*. For example, we may incompletely know the number and location of all people in the serviced area at each moment, and it is also uncertain when and why each of these people chooses to change location. Equivalently, in probability theory terms, the number of people who move between locations over time follows some latent (i.e., unknown) spatio-temporal distribution – a view

Figure 5.1: Our online predictive optimization framework. (a) Data about crowd movements is collected via WiFi probing. (b) The marginal distribution of each OD pair is estimated through Quantile Regression. (c) A copula combines the marginal distributions into a joint demand distribution. (d) Samples are drawn from the joint distribution. (e) An optimal route and corresponding frequency is determined for each sample. (f) The most frequently obtained optimal solution is selected for operation.

also expressed in, e.g., (Gonzalez et al., 2008), (Guo et al., 2012a), and (Guo and Karimi, 2017). At each lag then, a latent *joint* distribution accounts for the simultaneous movements between all OD pairs, each of which has its own latent *marginal* distribution of movements from Origin to Destination. The marginal distributions of different OD pairs may be correlated, and the joint distribution accounts for any such correlations.

The number of movements between each OD pair represents the *demand* for the mobility service, while the fleet of vehicles represents the *supply*. Based on the data collected at previous lags, our online framework predicts the demand, and accordingly optimizes the supply for the next lag, as illustrated in Figure 5.1. For demand prediction, the framework estimates the marginal distributions through Quantile Regression. For supply optimization, the framework decides an optimal design for the transit network using a scenario-based method. In each scenario, a draw is sampled from the predictive joint distribution of demand, and a corresponding linear programming instance is solved to obtain route structure and frequency for the sampled demand. Finally, the solutions for all samples are combined into an overall optimal network design for the next time lag.

This joint demand distribution is obtained by constructing a Gaussian copula, which is thus a key link between demand prediction and supply optimization in our framework. The use of a copula allows us to decouple these two components. Furthermore, as the copula preserves all correlations between OD pairs in the joint distribution, we are free to work with marginal demand distributions, which are easier to fit accurately for each OD pair.

Importantly, our framework retains the uncertainty in travel demand (rather than reduce it to point estimates), and takes advantage of this uncertainty during supply optimization. In this study, we advocate for preserving this uncertainty by estimating each marginal distribution through quantiles of its Cumulative Distribution Function. Another advantage of our framework is that our scenario-based approach to supply optimization yields a robust optimization scheme. That is, whereas most existing scenario-based optimization methods use predefined sets of scenarios, we generate scenarios dynamically and demand-responsively, based on real-time demand prediction.

5.1.2 Summary of Contributions

The main contributions of this paper are thus as follows:

- We develop a online predictive optimization framework for the demand-responsive transit network design problem, which decouples demand prediction from adaptive supply optimization, and integrates them through a copula.
- On the demand side, we develop several Quantile Regression models for predicting the marginal demand distribution for each Origin-Destination pair.
- On the supply side, we devise a robust optimization method for demand-responsive transit network design under stochastic demand, based on a novel minimum cost flow formulation.
- Using a case study based on real-world WiFi data, we demonstrate the capabilities of our predictive optimization framework, and show that it can outperform conventional optimization methods.

5.1.3 Paper Structure

The rest of this paper is structured as follows. In Section 5.2, we review literature on demand prediction and supply optimization under uncertainty, and

indicate the advantages of our study over previous works. In Section 5.3, we present different models for demand prediction with uncertainty, which we offer to do through Quantile Regression. There, we also apply the prediction models to a case study of autonomous mobility in a Danish university campus, based on actual WiFi records. In Section 5.4, we present a novel online method to determine the route structure and frequency for demand-responsive transit services under stochastic demand. In Section 5.5, we apply this stochastic optimization method to the aforementioned case study. Finally, in Section 5.6, we summarize our findings, draw conclusions, and identify future research directions.

5.2 Literature Review

In this Section, we review existing studies and state-of-the-art solution methods, while comparing them to our solution methodology. We begin by reviewing demand prediction with uncertainty, proceed to review supply optimization under demand uncertainty, and finally review methods for integrating both aspects.

5.2.1 Demand Prediction with Uncertainty

Accurate modeling of travel demand is essential for properly planning transit services: the more accurate the demand forecast, the better can service resources be allocated and scheduled ahead of time (Rodrigues and Pereira, 2020). Prediction of future demand for transport has thus been a long studied research topic, resulting in a plethora of parametric and non-parametric techniques for demand modeling (De Jong et al., 2007, Rasouli and Timmermans, 2012). However, despite the importance of accurate demand prediction, previous studies have often tended to over-simplify by providing only point estimates of future values (Rodrigues and Pereira, 2018). For example, transport studies often provide only the mean and standard deviation of the predictive distribution, either directly or, respectively, through the center and bounds of a confidence interval. Such point estimate methods, e.g., Maximum Likelihood Estimation and Maximum A-Posteriori, are commonly applied to a wide variety of demand modeling techniques (Shao et al., 2014, Tsai et al., 2009, Xue et al., 2015, Yang et al., 2013).

As the distribution of future demand may be highly irregular (e.g., skewed or multi-modal), reducing it to summary statistics may result in losing important information about the uncertainty structure of future demand. In turn, this can lead to inaccurate resource allocation and passenger dissatisfaction. Indeed, there are several benefits for preserving uncertainty in predictions, rather than

providing only point estimates (Li et al., 2019, Rodrigues and Pereira, 2018, Yang et al., 2019). On one hand, preserving uncertainty conveys a high degree of confidence in the predictions, so that corresponding decisions can be made more intelligently. For example, when given a full predictive distribution, a supply optimization method can prepare for a full range of possible scenarios, from best case to worst case. On the other hand, providing only point estimates might be misleading, e.g., the mean of a multi-modal distribution might in fact lie in a neighborhood of low probability.

To preserve uncertainty, Quantile Regression (QR) can be used to approximate a full predictive distribution by estimating several of its quantiles, without assuming any particular parametric form. This method has been applied to various problems in, e.g., econometric analysis (Chen, 2019), weather forecasting (Khan et al., 2019), and transport modeling (Rodrigues and Pereira, 2020). The QR model itself can follow either of several functional forms, such as linear or splines (Koenker, 2005), non-linear or non-parametric with Gaussian Processes (Antunes et al., 2017, Yang et al., 2018), or vector-valued (Sangnier et al., 2016). As more quantiles are used, the approximation which QR yields becomes more precise and more robust to artifacts in the true predictive distribution, such as multi-modality and non-symmetry.

In this paper, we evaluate the proposed framework through a case study, in which we estimate the distribution of future demand through Quantile Regression. Nevertheless, the framework supports multiple models other than QR for predicting arbitrary marginal distributions. For a moderately sized dataset as in our case study, Bayesian Inference (Peled et al., 2019c) can also be used with proper modeling of random variables and their dependency structure. Alternatively, Deep Neural Network models can be constructed along with multiple prediction intervals, as described in (Mazloumi et al., 2011) and (Khosravi et al., 2011).

5.2.2 Transit Network Design and Frequency Setting Problem (TNDFS)

The transit route planning process typically consists of five steps, as outlined in (Ceder and Wilson, 1986). The process is sequential, in that the decisions at each step become the input for the next step. The process starts with a network design problem, where the bus stops and routes are decided. Then the frequencies are determined based on the fleet size, followed by timetabling, vehicle scheduling, and finally the crew scheduling.

The transit network design and frequency setting problem (TNDFS) is a combination of two distinct sequential problems, i.e. the transit route network design

problem (TRNDP), which deals with the planning of optimal routes for transit services, and the frequency determination problem (FDP), which determines the frequencies for each of those routes. Extensive research into TRNDP variants and solution methods has been conducted since the late 1960s. We refer the readers to (Kepaptsoglou et al., 2009) for an extensive review of previous studies on TRNDP up to year 2007, in which the author classified the studies according to their objective functions, decision variables, transit network structure, demand patterns and characteristics, as well as the solution methods. We also provide references (Ceder, 2016, Farahani et al., 2013, Guihaire and Hao, 2008) that address various other aspects of the problem. Here, we focus the mathematical programming formulation for TRNDP under stochasticity.

5.2.2.1 Mathematical programming formulation for TRNDP

It is clear that while the solutions obtained at each step of the transit planning process are optimal, they may not necessarily be optimal for the overall transit planning problem. Despite the temptation to formulate a single model to globally optimize the transit planning procedure, the complexity involved should be taken into account, as the problem at each step of the process is highly combinatorial. The two steps relevant to our study are network design, and frequency determination, as we review next.

For network design, TRNDP instances are usually formulated based on a network graph, where the nodes represent transit stops, and edges represent connective paths between nodes. The objective is then to select which transit stops to serve, and the order of visiting them in each of the routes, based on travel demands and generalized costs. A solution method for network design gives the optimal routes and their corresponding temporal route length.

The frequency determination problem, on the other hand, is to optimally allocate vehicles to the different routes. This allocation largely depends on the temporal route lengths. The integration of the two problems can increase the complexity substantially, and often results either in a single highly nonlinear model (Cipriani et al., 2012b, Fan and Machemehl, 2008, Fan Wei and Machemehl Randy B., 2006b), or a bi-level mixed integer model (Szeto and Jiang, 2014, Szeto and Wu, 2011). The resulting formulations are complex, and rely on heuristics to solve for suboptimal solutions. In contrast, our optimization method uses a linear, multi-stage formulation, which can be solved through sampling, so that the solution is optimal in expectation.

5.2.2.2 Stochastic Transit Network Design Problem with Uncertain Demands

In the literature, TRNDP is usually formulated as a *static* linear program with parameters that take on deterministic values. In practice, however, these parameters are often not static but *stochastic*, so that they follow some probability distributions.

While some previous studies have considered demand elasticity, which is an inherent property of a real transit network, most studies have only considered fixed demand. In (Fan Wei and Machemehl Randy B., 2006b), Fan and Machemehl have attributed this to the NP-hard complexity of TRNDP. The consideration of elastic demands often results in an iterative procedure, which repeatedly chooses routes structure and demand splits until some convergence criterion is achieved (Cipriani et al., 2012a, Fan Wei and Machemehl Randy B., 2006a, Lee Young-Jae and Vuchic Vukan R., 2005). However, this would mean that no optimal solution can be guaranteed due to the heuristic nature of the problem. Therefore, in our framework, demands are not considered as directly dependent on the service quality. Instead, the elasticity is internalized in the demand stochasticity alongside other factors, e.g., weather and seasonal variations.

Approaches for dealing with such stochasticity commonly reduce the distributions to point estimates. One such approach is to use the expected values of parameter distributions (Cipriani et al., 2012b, Fan and Machemehl, 2008). While the expected value formulation may be simple to obtain, its solution nonetheless lacks robustness. To overcome this weakness, minimax robust optimization can be used (An and Lo, 2015a, 2016, Laporte et al., 2010, Lou et al., 2009), whereby the maximum of the support of the parameters is taken instead of their expected values. Doing so ensures that the solution is feasible for all possible combinations of parameters. This approach is often called the *worst-case scenario* approach, because the solution space encapsulates all possible combinations, including the worst-case scenario.

Nevertheless, there are drawbacks to both approaches for dealing with stochasticity in parameters. First, by reducing the parameter distributions to point values, both approaches discard of useful information in the full distributions. Second, while minimax robust optimization guarantees feasibility for all possible demand scenarios, it can sometimes be overly conservative, as acknowledged in (An and Lo, 2015a). In contrast, we optimize service routes and frequencies based on a full estimate of the predictive demand distribution, rather than point estimates.

Another commonly used method to improve robustness is scenario-based robust optimization. There, stochasticity is generated either by adding random perturbations to the average demands, by explicitly constructing demand scenarios for

different seasons and/or time of the day, or by random sampling of parameters from their probability densities. For instance, in (Amiripour S. M. Mahdi et al., 2014) Amiripour et al. consider TRNDP with variable demands. To simulate demand stochasticity, they generate 480 perturbations of a demand matrix, each by adding stochastic noise. While this method does add robustness to the solution by introducing random noise in the demand, the random perturbations may not necessarily reflect the stochasticity that can be observed in collected demand data.

In contrast, the optimization method we devise in this study takes advantage of the full predictive distributions of parameters. As (stochastic) TRNDP is an NP-hard combinatorial problem, solution methods for TRNDP are often approximate, and so is our method. Specifically, our solution method for stochastic TRNDP is based on sampling from the distribution of parameters, so that each sample is a static problem instance, uniquely defined by the drawn parameter values. Consequently, the solution that our method yields is optimal in expectation.

5.2.3 Combining Demand Prediction with Supply Optimization

Several studies cater for both aspects of demand-responsive supply optimization, i.e., explicitly provide both a model of future demand and an algorithm for demand-responsive supply optimization. The methods that these studies develop for combining both aspects generally fall into two categories: hybrid and decoupled. Hybrid methods (e.g., (Cortés et al., 2009, Powell and Topaloglu, 2003, Stein, 1978)) combine demand prediction and supply optimization within one formulation, often through stochastic programming. Conversely, decoupled methods (e.g., (Alonso-Mora et al., 2017, Ferrucci et al., 2013, Iglesias et al., 2018)) model each of the two components separately, so that supply is optimized based on the output of an independent demand model.

As noted in (Ichoua et al., 2006), notable works on hybrid methods include a line of papers by Powell and his team, dating back as early as 1988 ((Godfrey and Powell, 2002, Powell, 1988, 1996, Powell and Topaloglu, 2003, Spivey and Powell, 2004)). A more recent example of the hybrid approach is by Cortes et al. (Cortés et al., 2009) (2009), who study a delivery-and-pickup service with predetermined stations, and develop a state-space model where the objective function to be optimized consists of both demand estimate and route cost. Our solution framework, however, promotes a decoupled approach, which allows us to model each component independently, while joining them through a copula (as defined later in Sec. 5.4.1). This decoupling allows us to freely compare several models with uncertainty estimates on the demand prediction side, while offering

a non-myopic algorithm on the supply optimization side, i.e., an algorithm that considers multiple future scenarios to improve user gains. The use of a copula further allows us to derive marginal demand distributions for each OD pair, rather than estimate an entire joint distribution at once.

A common deficiency in works about decoupled methods is that the unified solution they propose relies on an possibly overly simplified demand prediction component. Such is the case in the seminal work by Stein (Stein, 1978) (1978), who studies route optimization for Dial-a-Ride services, while assuming that users' requests are independently drawn from a uniform spatial distribution over the serviced area. Ferrucci et al. (Ferrucci et al., 2013) (2013) present another approach to demand-responsive supply optimization, whereby the service area is segmented, and each segment is assumed to follow a Poisson distribution with time-dependent rate. Iglesias et al. (Iglesias et al., 2018) (2018) offer a data-driven framework to rebalance an autonomous on-demand fleet, where demand is predicted based on a Deep Neural Network model, which does not yield uncertainty estimates. While our framework also takes a decoupled approach, it does not presume a particular form of demand distribution, and can rather approximate it through e.g., Quantile Regression (QR, as defined in Sec. 5.3.2). This is particularly applicable to the case study of mobility in a university campus, where student movements are often spatio-temporally correlated and unevenly distributed.

Let us conclude the literature review with two recent (2017) investigations that bear particular resemblance to ours. The first is by Alonso-Mora et al. (Alonso-Mora et al., 2017), where the authors study urban-scale route optimization for a hypothetical fleet of self-driving taxis. Similarly to our approach, they too leverage historical data for real-time demand prediction, and construct a marginal probability distribution for each OD pair. Nevertheless, their method relies on the frequentist approach, which can be beneficial for large datasets, but fails to retain enough uncertainty for smaller datasets as in our case.

The second study which particularly resembles ours is by Miller et al. (Miller and How, 2017), where the authors too predictively optimize a small fleet of on-campus autonomous shuttles, with the objective of minimizing expected customer waiting time. However, whereas they focus on predicting key locations for proactive positioning of the vehicles, our objective is to predictively optimize service routes and frequencies. Also, their sources of demand information are users' requests via a dedicated smartphone app, and sensors that the few vehicle carry around. In contrast, our source of demand information is a network of hundreds of WiFi hotspots, fixed all over campus, which thus provide high spatio-temporal observability. Finally, whereas our framework offers online predictive optimization, their method is offline, and they differ online demand prediction to future work.

Figure 5.2: Data in our case study: (5.2a) Observed locations in DTU Lyngby Campus, each covering one or more buildings. (5.2b) Observed hourly movements between all locations, with dates specified for Mondays.

5.3 Demand Prediction through Quantile Regression

In this Section, we describe how Quantile Regression (QR) can be applied to demand prediction with uncertainty. We demonstrate the application of QR step-by-step through a case study of predictive optimization for an autonomous shuttle service in a Danish University campus as part of a real-world project (LINC, 2020). For this project, we obtained a dataset of WiFi records, collected from various buildings on the campus for several weeks in 2017 and 2018. We also use this case study in later Sections to describe and test our stochastic optimization method.

5.3.1 Data for Demand Estimation

The data in this case study consists of crowd movements between 6 different locations in the Lyngby campus of the Technical University of Denmark (DTU), illustrated in Figure 5.2a. We label these locations as $g_{10}, g_{11}, g_{20}, g_{30}, g_{37}, g_{42}$ per internal building numbering in DTU, hence label numbers are not necessarily consecutive. In each location, WiFi access points probe for wireless devices, and we are given hourly aggregated counts of wireless devices that change location. In total, we have $6 \cdot 6 - 6 = 30$ time series of hourly aggregated movements, one for each OD pair, ranging from 17-Nov-2017 00:00 to 14-Jan-2018 23:00.

The aggregated counts contain no information about individual devices. We use the counts of wireless devices as a proxy for counts of people on campus, and assume that approximation errors amount to systematic noise in the observations. For example, we assume that each person carries a personally fixed number of wireless devices (phone, laptop, tablet, etc.), and thus increases the

approximated total by a consistent overhead.

5.3.2 Quantile Regression

As explained in Section 5.1.1, the total number of movements from each Origin to each Destination at each lag follows a latent marginal distribution. The density of this distribution can be estimated through quantiles of its Cumulative Distribution Function (CDF), as follows.

For any OD pair and lag t , let y_t denote the observed number of movements from origin to destination at lag t . y_t is thus a realization from a random variable Y_t , which has the corresponding latent marginal distribution of total movements. Next, for any $0 < q < 1$, let $y_t^{(q)}$ denote the q 'th quantile of the CDF of Y_t , i.e., the smallest real that the CDF maps to q . We shall estimate $y_t^{(q)}$ for each $q \in Q = \{5\%, 25\%, 50\%, 75\%, 95\%\}$, and so obtain the following approximation of the marginal distribution:

$$\Pr\left(0 \leq y_t \leq \hat{y}_t^{(0.05)}\right), \Pr\left(\hat{y}_t^{(0.05)} < y_t \leq \hat{y}_t^{(0.25)}\right), \\ \dots, \Pr\left(\hat{y}_t^{(0.75)} < y_t \leq \hat{y}_t^{(0.95)}\right), \Pr\left(\hat{y}_t^{(0.95)} < y_t\right), \quad (5.1)$$

where $\hat{y}_t^{(q)}$ is our estimation of the true $y_t^{(q)}$. This manner of density estimation is thus named "Quantile Regression" (QR). To illustrate, Figure 5.3 on the facing page shows an example of $\hat{y}_t^{(q)}$ for one of our QR models, which we define later.

Before modeling, we need to treat a couple of issues in the time series, which might adversely affect model performance. First, for some of the OD pairs, the time series of observations is non-stationary. Second, for all OD pairs, there are nearly no movements on campus during 23:00...06:59 and during Christmas holiday, [23-Dec-2017, ..., 1-Jan-2018]. We thus transform the time series of each OD pair in two steps. First, we difference the time series as $\{y'_t\} = \{y_t - y_{t-1}\}$, which is stationary for all OD pairs, as we verify through augmented Dickey-Fuller test with $p\text{-value} = 1\%$. Second, we remove from $\{y'_t\}$ all lags in hours 23:00...06:59 (these can also be used for shuttle maintenance) and in the Christmas holiday.

We are now ready to fit and test various demand prediction models on the remaining lags in the transformed time series, which we partition as follows: train set T_{train} consists of all remaining lags in 17-Nov-2017, ..., 7-Jan-2018 (52 days), whereas test set T_{test} consists of all other remaining lags in 8-Jan-2018, ..., 14-Jan-2018 (7 days), as illustrated in Figure 5.2b on the previous page. We train each QR model on T_{train} , and then evaluate it on T_{test} by calculating the following measures for each OD pair:

Figure 5.3: Example output of a Quantile Regression model (LQR_3^{Ind}). Top: estimated quantiles for every OD pair at one lag ($t = 2018-01-08$ 20:00), where y axes are logarithmic between 0 and 200. Bottom: estimated quantiles for the entire test set of one OD pair ($g_{42} \rightarrow g_{10}$).

- Mean Titled Loss:

$$\text{MTL} = \sum_{q \in Q} \frac{1}{|T_{test}|} \sum_{t \in T_{test}} \max \{q(y_t - \hat{y}_t^{(q)}), (q-1)(y_t - \hat{y}_t^{(q)})\}.$$

- Percent of measured values that fall within the 5% – 95% quantile range:

$$\text{ICP}_{5-95} = \frac{1}{|T_{test}|} \left| \left\{ t \in T_{test} \mid \hat{y}_t^{(0.05)} \leq y_t \leq \hat{y}_t^{(0.95)} \right\} \right|.$$

- Mean length of 5% – 95% quantile range:

$$\text{MIL}_{5-95} = \frac{1}{|T_{test}|} \sum_{t \in T_{test}} \hat{y}_t^{(0.95)} - \hat{y}_t^{(0.05)}.$$

- Number of pairwise quantile crossings:

$$\# \text{cross} = \sum_{t \in T_{test}} \left| \left\{ (q_i \in Q, q_j \in Q) \mid q_i < q_j \wedge \hat{y}_t^{(q_i)} > \hat{y}_t^{(q_j)} \right\} \right|.$$

Model	Total MTL	Mean ICP ₅₋₉₅	Mean MIL ₅₋₉₅	Mean #cross
HP^{Ind}	446.071	0.731 (± 0.051)	26.129 (± 32.929)	0.000 (± 0.000)
LQR_1^{Ind} w. Seasonality	437.805	0.879 (± 0.055)	38.741 (± 48.215)	13.800 (± 20.908)
LQR_2^{Ind} w/o Seasonality	459.428	0.777 (± 0.055)	31.287 (± 39.255)	15.667 (± 9.321)
LQR_3^{Ind} w. Sorting	437.218	0.882 (± 0.053)	38.781 (± 48.265)	0.000 (± 0.000)
LQR_4^{Ind} w. Exams	437.235	0.900 (± 0.049)	39.692 (± 49.696)	0.000 (± 0.000)
LQR_5^{Ind} w/o first week	444.537	0.896 (± 0.045)	40.463 (± 50.574)	0.000 (± 0.000)
DNN^{Ind} FC Linear, Common LR	475.826	0.925 (± 0.041)	42.043 (± 55.191)	0.000 (± 0.000)
DNN^{Ind} FC Linear, LR per OD	474.174	0.930 (± 0.035)	41.384 (± 50.659)	0.000 (± 0.000)
DNN^{Ind} FC Linear, LR per OD and q	477.845	0.949 (± 0.034)	52.220 (± 69.943)	0.000 (± 0.000)
$GBoost^{Ind}$, Common Params	455.074	0.839 (± 0.042)	40.041 (± 56.008)	0.000 (± 0.000)
$GBoost^{Ind}$, Params per OD	455.295	0.838 (± 0.050)	39.037 (± 55.906)	0.000 (± 0.000)
$GBoost^{Ind}$, Params per OD and q	446.999	0.846 (± 0.041)	39.508 (± 56.020)	0.000 (± 0.000)
LQR_1^{Mul} All ODs Together	494.227	0.918 (± 0.104)	41.663 (± 48.100)	0.000 (± 0.000)
LQR_2^{Mul} similar to $VAR(1)$	485.963	0.876 (± 0.046)	39.421 (± 50.775)	0.000 (± 0.000)
DNN_1^{Mul} All Quantiles Together	468.709	0.907 (± 0.047)	36.431 (± 43.771)	0.000 (± 0.000)
DNN_2^{Mul} Parameter Sharing	777.361	0.852 (± 0.072)	45.634 (± 66.715)	0.000 (± 0.000)
$GBoost_1^{Mul}$ Common Params	433.980	0.883 (± 0.040)	34.621 (± 39.883)	0.000 (± 0.000)
$GBoost_2^{Mul}$ Params per q	437.528	0.884 (± 0.046)	37.649 (± 43.113)	0.000 (± 0.000)

Table 5.1: Predictive performance of Quantile Regression models. The best performing model is highlighted.

Because MTL is a weighted mean of prediction errors, a QR prediction model is considered better if it achieves lower total MTL over all OD pairs. The other measures are averaged for each model over all OD pairs, and we use them as additional indicators of performance quality: it is preferred to bring ICP₅₋₉₅ closer to $95\% - 5\% = 90\%$ while simultaneously obtaining lower mean MIL₅₋₉₅, and it is also preferred to obtain fewer quantile crossings.

For the remainder of this Section, we define and experiment with the following types of Quantile Regression models: Historical Percentiles, Linear QR, Deep QR, and Gradient Boosting QR. Table 5.1 summarizes the overall predictive performance of all models, and highlights the best performing model.

5.3.3 Independent Models

We begin by fitting models independently for each $q \in Q$ and each OD pair. For each model type, we thus fit a total of $5 \cdot 30 = 150$ independent models, and evaluate their overall performance as explained above. We use superscript Ind as part of a model name (e.g., as in HP^{Ind}) to indicate such independent fitting.

5.3.3.1 Modeling by Historical Percentiles

For any lag t , let $TOD(t) \in \{7, \dots, 22\}$ denote the time-of-day of t , and let $DOW(t) \in \{0, \dots, 6\}$ denote the day-of-week of t , so that $0 \equiv \text{Monday}$. Our first model is a naive Historical Percentiles model HP^{Ind} , in which for each OD pair and each $q \in Q$, $\hat{y}_t^{(q)}$ is the q 'th percentile of

$$\{y_k \mid k < t \wedge DOW(k) = DOW(t) \wedge TOD(k) = TOD(t)\}. \quad (5.2)$$

As Table 5.1 shows, HP^{Ind} achieves ICP_{5-95} far from 90%, which indicates that the underlying time series is not a simple repetition of historical patterns. Moreover, if instead of Q we fit HP^{Ind} on the 0%'th and 100%'th quantiles – namely the minima and maxima in the train set – we obtain mean ICP only 80% ($\pm 5.5\%$), which is far from 100%. This means that for some OD pairs, there are movement counts in the test set that lie outside of the range of values in the train set, which confirms that historical percentiles alone are insufficient predictors for this data.

5.3.3.2 Linear Quantile Regression

We proceed to more flexible Linear Quantile Regression (LQR) models, where the estimated number of movements is modeled as follows:

$$\begin{aligned} \hat{y}_t^{(q)} = & \beta_7^{TOD} d_t^{(7)} + \dots + \beta_{22}^{TOD} d_t^{(22)} \\ & + \beta_0^{DOW} w_t^{(0)} + \dots + \beta_6^{DOW} w_t^{(6)} \\ & + \beta_{-1} y_{t-1} + \dots + \beta_{-24} y_{t-24}, \end{aligned} \quad (5.3)$$

where $d_t^{(i)} \in \{0, 1\}$ is 1 iff $i = TOD(t)$, $w_t^{(i)} \in \{0, 1\}$ is 1 iff $i = DOW(t)$, and the parameters to be estimated are β 's (with subscripts corresponding to TOD , DOW , and lag). As for all other QR prediction models, the loss function to be minimized is the total MTL . We train all LQR models via Iterative Weighted Least Squares with an Epanechnikov kernel and Hall-Sheather bandwidth selection (Hall and Sheather, 1988), and clip any negative predictions to zero.

Our first linear model LQR_1^{Ind} achieves better total MTL and significantly better mean ICP_{5-95} than HP^{Ind} . Next, we attempt to further improve performance by removing seasonality from the data. That is, we transform y'_t into $y''_t = (y'_t - \bar{y}'_t) / \tilde{y}'_t$, where $\bar{y}'_t$ and $\tilde{y}'_t$ are, respectively, the mean and standard deviation of $\{y'_k \mid k \in T_{train} \wedge DOW(k) = DOW(t) \wedge TOD(k) = TOD(t)\}$. However, LQR_2^{Ind} performs worse on $\{y''_t\}$ than LQR_1^{Ind} does on $\{y'_t\}$.

We thus proceed to run LQR again on $\{y_t'\}$, and this time eliminate quantile crossings by sorting $\hat{y}_t^{(0.05)}, \dots, \hat{y}_t^{(0.95)}$ in ascending order for every t . This results in a better performing model LQR_3^{Ind} , for which we have illustrated some predictions earlier in Figure 5.3 on page 103.

During [8-Dec-2017, ..., 22-Dec-2017], exams took place on campus. Hence we next add to each data vector a binary feature m_t , which indicates whether the data is in the exam period, namely:

$$\begin{aligned} \hat{y}_t^{(q)} = & \beta_7^{TOD} d_t^{(7)} + \dots + \beta_{22}^{TOD} d_t^{(22)} \\ & + \beta_0^{DOW} w_t^{(0)} + \dots + \beta_6^{DOW} w_t^{(6)} \\ & + \beta^{EX} m_t \\ & + \beta_{-1} y_{t-1} + \dots + \beta_{-24} y_{t-24}. \end{aligned} \quad (5.4)$$

The corresponding model is LQR_4^{Ind} , which we train with quantile sorting. LQR_4^{Ind} yields nearly the same total MTL as does LQR_3^{Ind} , and mean ICP_{5-95} noticeably closer to 90%, with a small increase in mean MIL_{5-95} . We thus designate LQR_4^{Ind} as the best performing model so far.

We next proceed to Deep Quantile Regression, where the training process requires us to reserve a week of the data for validation. Hence for fair comparison with LQR, we also build and estimate model LQR_5^{Ind} , where we omit the first week from the train set. Consequently, LQR_5^{Ind} performs worse than LQR_4^{Ind} , where the train set includes the first week.

5.3.3.3 Deep Quantile Regression

Whereas LQR models are parametric and linear, Deep Neural Network (DNN) models provide a non-parametric approximation of the true predictive distribution. As a baseline, we wish to first obtain similar performance for DNN as for LQR , hence our first deep QR model DNN^{Ind} is as in Figure 5.4 on the facing page: a fully-connected, feed-forward neural network, where a single linear unit combines all input features. While training DNN^{Ind} , we use the first week of T_{train} as a validation set for early stop. We thus aim for DNN^{Ind} to first perform as well as LQR_5^{Ind} , where we have omitted the first week from the train set too.

The performance of DNN strongly depends on hyper-parameter selection, hence before training, we tune a learning rate in $[10^{-8}, 1]$ using Bayesian Optimization: a common approach for hyper-parameter tuning (Snoek et al., 2012). To this end, we partition T_{train} into three subsets: the first 7 days in T_{opt}^{test} for testing, the following 7 days in T_{opt}^{val} for validation, and the remaining days in T_{opt}^{train} for

Figure 5.4: DNN^{Ind} , where $x_{t,1}, \dots, x_{t,n}$ are input features at lag t , and $\beta_1^{(q)}, \dots, \beta_n^{(q)}$ are trainable parameters.

training. In every iteration, the optimizer tries a different learning rate to train the DNN on T_{opt}^{train} , while using T_{opt}^{val} for early stop, and then obtains the total MTL of the trained DNN on T_{opt}^{test} . The optimal learning rate is the one which minimizes this total MTL after at most 100 iterations.

We note again that learning rate optimization is done separately of and prior to actually fitting the parameters of the DNN models. That is, we use data sets $T_{opt}^{val}, T_{opt}^{train}, T_{opt}^{test}$ only during learning rate optimization, whereas during actual model fitting, we use T_{train} and T_{test} as for all other demand models (while internally using the first week of T_{train} to avoid overfitting). We also note that while the same learning rate may be common to several models, we still fit every DNN^{Ind} model independently for each $q \in Q$ and each OD pair.

Figure 5.5 on the next page describes the learning rate optimization for independent Deep QR models. First, we optimize a single learning rate, common to all OD pairs and all $q \in Q$ (Figure 5.5a on the following page). The common learning rate yields worse performance than LQR_5^{Ind} , as shown in Table 5.1 on page 104. We thus next optimize a learning rate independently for each OD pair (Figure 5.5b on the following page). This results in a little improvement over the common learning rate, but performance is still noticeably worse than LQR_5^{Ind} . Finally, we optimize the learning rate independently for every combination of OD and $q \in Q$ (Figure 5.5c on the next page), but gain no performance improvement.

In summation, Deep Quantile Regression did not perform as well as Linear Quantile Regression, despite learning rate optimization. The dataset is thus not large enough for effective deep learning with backpropagation, hence we do

Figure 5.5: Bayesian Optimization of learning rate (LR) of DNN^{Ind} . The best LR, indicated by a vertical line, can shift considerably for the same OD pair under different optimization resolutions.

not attempt to further improve performance, e.g., using kernel regularization, dropout, or recurrent units. As an alternative attempt at non-parametric QR, we next experiment with Gradient Boosting.

5.3.3.4 Gradient Boosting

Gradient Boosting (GBoost) (Friedman, 2002) builds an ensemble of regression trees incrementally using gradient descent. As in previous prediction models, the loss function being boosted is total MTL . Figure 5.6 on the facing page shows an example of $GBoost^{Ind}$ for one OD pair and a single $q \in Q$.

As for DNN^{Ind} , we partition T_{train} as $T_{opt}^{train}, T_{opt}^{val}, T_{opt}^{test}$, and use Bayesian Optimization to find the best hyper-parameters for $GBoost^{Ind}$: learning rate in $[10^{-8}, 1]$, maximum tree depth in $1..6$, and maximum number of trees in $1..200$. Figure 5.7b on the next page shows that for all $q \in Q$, the optimal learning for $GBoost^{Ind}$ follows a similar pattern of distributions as for DNN^{Ind} in Figure 5.7a on the facing page, albeit with greater variance for $q \in \{0.25, 0.75\}$. Nevertheless, Table 5.1 on page 104 shows that contrary to DNN^{Ind} , the performance of $GBoost^{Ind}$ improves when hyper-parameter optimization is carried out separately for each OD and $q \in Q$. Compared to LQR_5^{Ind} , the best performing $GBoost^{Ind}$ has similar total MTL and mean MIL_{5-95} , but worse mean ICP_{5-95} .

Figure 5.6: A Gradient Boosting model passes the input $x_t = (x_{t,1}, \dots, x_{t,n})$ through each decision tree until reaching a leaf, then outputs the sum of all these leaves. Edge labels correspond to decision rules, which we arbitrarily instantiate here for demonstration purposes.

Figure 5.7: Boxplot of optimal learning rate distribution over all OD pairs, for each $q \in Q$.

5.3.4 Modeling Spatio-Temporal Dependencies

For every type of prediction model above, we have built 150 independent models, each pertaining to a different OD pair and $q \in Q$. Let us now examine models which process together multiple OD pairs or multiple quantiles, and so may be able to exploit spatio-temporal dependencies in the data. Indeed, Figure 5.8 shows that for some couples of OD pairs, the time series of movements are strongly correlated.

Figure 5.8: Pearson correlation between the time series of every couple of different OD pairs.

5.3.4.1 Linear Quantile Regression on Multiple Time Series

We begin with a linear QR model LQR_1^{Mul} which processes all 30 OD pairs together, independently for each $q \in Q$, as:

$$\begin{aligned}
 \hat{y}_t^{(q)} = & \beta_7^{TOD} d_t^{(7)} + \dots + \beta_{22}^{TOD} d_t^{(22)} \\
 & + \beta_0^{DOW} w_t^{(0)} + \dots + \beta_6^{DOW} w_t^{(6)} \\
 & + \beta^{EX} m_t \\
 & + \beta_{-1} y_{t-1} + \dots + \beta_{-24} y_{t-24} \\
 & + \beta_1^{OD} o_1 + \dots + \beta_{30}^{OD} o_{30},
 \end{aligned} \tag{5.5}$$

where for an arbitrary, pre-fixed ordering of the 30 OD pairs, o_i is binary and indicates whether the data corresponds to the i 'th OD pair. We train and test LQR_1^{Mul} using the same methods as for LQR_4^{Ind} . Table 5.1 on page 104 summarizes the performance of LQR_1^{Mul} , which is seen to be significantly worse than LQR_4^{Ind} .

Figure 5.9: Multivariate deep Quantile Regression models for any OD pair. $x_{t,1}, \dots, x_{t,n}$ are input features at lag t .

Next, we try linear model LQR_2^{Mul} , where for each OD pair independently:

$$\begin{aligned}
 \hat{y}_t^{(q)} = & \beta_7^{TOD} d_t^{(7)} + \dots + \beta_{22}^{TOD} d_t^{(22)} \\
 & + \beta_0^{DOW} w_t^{(0)} + \dots + \beta_6^{DOW} w_t^{(6)} \\
 & + \beta^{EX} m_t \\
 & + \sum_{i=1}^{30} \sum_{k=1}^p \beta_{-k}^{(i)} y_{t-k}^{(i)},
 \end{aligned} \tag{5.6}$$

where $p \in \mathbb{N}$ is a hyper-parameter, and $y_{t-k}^{(i)}$ is the k 'th past lag of the i 'th OD pair. LQR_2^{Mul} is therefore similar to $VAR(p)$, namely Vector Autoregression of order p , where each of several response variables is modeled on the last p lags of all response variables. We have experimented with $p = 1, 2, 3, 4$, and noticed that performance deteriorated as p increased. Table 5.1 on page 104 summarizes the performance of LQR_2^{Mul} for $p = 1$, which is seen to perform better than LQR_1^{Mul} , but still worse than LQR_4^{Ind} .

5.3.4.2 Multivariate Deep Quantile Regression

Next, we try multivariate variants of the DNN models which we built earlier. These variants process multiple quantiles together, and so take less time to train and optimize than the earlier univariate, independent models.

Our first deep multivariate model DNN_1^{Mul} is as illustrated in Figure 5.9a, such that for each OD pair independently, all quantiles are modeled together. We

again partition T_{train} as $T_{opt}^{train}, T_{opt}^{val}, T_{opt}^{test}$, and use Bayesian Optimization to find the best learning rate in $[10^{-8}, 1]$, common to all OD pairs. The performance of DNN_1^{Mul} in Table 5.1 on page 104 is seen to be worse than LQR_5^{Ind} , but better than the univariate DNN^{Ind} models.

Modeling multiple quantiles together can thus result in improved performance. We next try to further improve performance by introducing a hidden layer between the input and the output layers, as in Figure 5.9b on the preceding page, so that different quantiles share some trainable parameters. Table 5.1 on page 104 shows that this model, which we name DNN_2^{Mul} , performs badly.

5.3.5 Gradient Boosting on Multiple Time Series

Finally, we fit Gradient Boosting (GBoost) models on the time series of all OD pairs together, independently for each $q \in Q$. As in Section 5.3.3.4, we use Bayesian Optimization to find the best hyper-parameters, and summarize results in Table 5.1 on page 104.

First, we optimize a single set of hyper-parameters, common to all $q \in Q$. This results in model $GBoost_1^{Mul}$, which outperforms LQR_4^{Ind} , our previously best performing prediction model. That is, compared to LQR_4^{Ind} , $GBoost_1^{Mul}$ achieves better total MTL and significantly better mean MIL_{5-95} , with a small decrease in mean ICP_{5-95} . Next, we optimize hyper-parameters independently for each $q \in Q$. The resulting model $GBoost_2^{Mul}$ achieves similar total MTL and better mean MIL_{5-95} vs. LQR_4^{Ind} , but does not improve over $GBoost_1^{Mul}$. In summation, $GBoost_1^{Mul}$ yields the best prediction quality among all QR models.

5.3.6 Conclusions on Demand Prediction with Quantile Regression

We have experimented with four types of QR models for predicting distributions of future demand: naive Historical Percentiles, Linear QR, Deep QR, and Gradient Boosting QR. Our experiments include independent models for each OD pair and each $q \in Q$, as well as multivariate models which simultaneously process multiple OD pairs or multiple quantiles. Table 5.1 on page 104 summarizes the experiment results.

Historical Percentiles performs rather poorly, which indicates that the data at hand does not follow a simple repeating pattern. For Linear and Deep QR, independent models mostly yield better prediction quality than multivariate models. For Gradient Boosting QR, however, multivariate models outperform independent models, such that $GBoost_1^{Mul}$ is the overall best performing prediction

model.

In conclusion, model $GBoost_1^{Mul}$ can exploit spatio-temporal dependencies in the data to some extent. Indeed, Figure 5.8 on page 110 shows that for some OD pairs, the time series of movement counts are strongly correlated.

5.3.7 Online Demand Prediction

The models we have developed for demand prediction are data-driven, and can be trained offline on historical data. Given online data for recent lags, it takes only a few seconds to apply the trained models and obtain predictive distributions for the next lag. As new data accumulates over time, the models can be re-trained offline, e.g., once every few days.

5.4 Supply Optimization

So far, we have described how our framework yields predictive distributions of marginal demand. In this Section, we devise a method for transit network design under such demand stochasticity. As explained in 5.2.2.2, this moves us from the context of the classic, static TRNDP, where demands take on deterministic values, to stochastic TRNDP, where demands follow some distributions.

Our solution approach to *stochastic* TRNDP is scenario-based: by independently drawing values for the parameters from their distributions, we generate multiple *static* TRNDP instances, which we then solve independently. Each solution provides an optimal route and service frequency for the corresponding instance. Finally, we combine all solutions into an overall optimal solution.

Sampling from the *joint* distribution of parameters is made possible by constructing a Gaussian copula, which can join the *marginal* distributions. In the following Sections, we first explain how to build the copula, then describe our formulation of the optimization problem instances, and finally describe how to combine them into an overall solution, which is optimal in expectation. These steps are summarized in Algorithm 2.

5.4.1 Sampling via Gaussian Copula

To obtain multiple instances of the stochastic TRNDP problem, we need to be able to jointly sample from the marginal distributions of the parameters.

Algorithm 2: Supply optimization at lag t

Input: Marginal predictive distributions $\hat{y}_{t,OD_1}, \dots, \hat{y}_{t,OD_N}$ (Sec. 5.3.2) ;
 copula H (Sec. 5.4.1.1) ; number of samples k .
 Output: Optimal solution X_t^* .

- 1 Using H , draw k independent samples $s_{t,1}, \dots, s_{t,k}$ from the joint distribution of $\hat{y}_{t,OD_1}, \dots, \hat{y}_{t,OD_N}$ (Sec. 5.4.2.3).
- 2 foreach $s_{t,i}$ in $s_{t,1}, \dots, s_{t,k}$ do
- 3 Calculate optimal solution $X_{t,i}^*$ by solving the linear program in Sec. 5.4.2.2
 for $s_{t,i}$.
- 4 return the most frequent element among $X_{t,1}^*, \dots, X_{t,k}^*$.

In order to achieve joint sampling of the distributions, we require a function that pieces together the individual distributions to form a joint distribution. The function, in essence, retains the correlation structure between the different parameters, and provides us with a mean of obtaining joint samples. We next define the Gaussian copula and describe how our framework uses it for stochastic optimization.

5.4.1.1 Gaussian Copula

Let $F_1(x), \dots, F_n(x)$ be the marginal cumulative distribution functions (CDFs) of demand parameters for each OD pair $X_1, \dots, X_n$, respectively. The random vector $(X_1, \dots, X_n)$ is distributed per the joint cumulative distribution function (CDF) of its components:

$$H(x_1, \dots, x_n) = \Pr(X_1 \leq x_1, \dots, X_n \leq x_n). \quad (5.7)$$

If each of $F_1(x), \dots, F_n(x)$ is continuous, then by Sklar's theorem, H is uniquely defined as a function of these marginal CDFs, namely

$$H(x_1, \dots, x_n) = C(F_1(x_1), \dots, F_n(x_n)). \quad (5.8)$$

C is then called a *copula*, and by using this formulation, we decouple the correlation structure in $(X_1, \dots, X_n)$ from the individual marginals. A commonly used type of copula is the Gaussian copula,

$$C_G(u_1, \dots, u_n) = \Phi(u_1, \dots, u_n; \Sigma), \quad (5.9)$$

where Φ is the joint CDF of the zero-mean multivariate normal distribution with covariance matrix Σ .

We can use the Gaussian copula to obtain samples from the joint distribution of $(X_1, \dots, X_n)$. To obtain such a sample, we first draw $s = (s_1, \dots, s_n)$ from C_G (Eq. 5.9). Then, using the multivariate normal CDF Φ_Σ , we transform s into

$\phi = (\Phi_\Sigma(s_1), \dots, \Phi_\Sigma(s_n))$. Finally, we map ϕ to the desired sample through the inverses of the marginal CDFs, as

$$\left(F_1^{-1} \Phi_\Sigma(s_1), \dots, F_n^{-1} \Phi_\Sigma(s_n)\right). \quad (5.10)$$

5.4.1.2 Covariance Matrix for the Gaussian Copula

In this study, we assume that only travel demands are stochastic, while other parameters, such as travel time between nodes, remain constant over time. To construct the copula, we assume for simplicity that the correlation structure of the data is time-invariant. As such, it suffices to compute the covariance matrix of the copula offline once, based on historical data. We note, however, that this computation can be efficiently maintained online as new data becomes known, so that the copula retains the updated state of correlation.

To construct the covariance matrix for the Gaussian copula C_G in our framework, we use the observed movement counts in T_{train} , as following. First, for the n demand nodes in the TRNDP graph G , we calculate for each OD pair its empirical marginal CDF F_{od} , based on all its historically observed movement counts in T_{train} . For each $t \in T_{train}$, we then collect the corresponding observations of all OD pairs into a travel demand vector of size n^2 ,

$$\mathbf{d}^{(t)} = \left(d_{11}^{(t)}, \dots, d_{1n}^{(t)}, d_{21}^{(t)}, \dots, d_{2n}^{(t)}, \dots, d_{n1}^{(t)}, \dots, d_{nn}^{(t)}\right), \quad (5.11)$$

where the travel demand from any demand node to itself, i.e. d_{ii} for $i = [1, n]$, is 0. Next, we transform $\mathbf{d}^{(t)}$ element-wise into the Gaussian layer as follows:

$$\tilde{d}_{od}^{(t)} = \begin{cases} \Phi^{-1} F_{od} \left(d_{od}^{(t)}\right), & o \neq d \\ 0, & o = d \end{cases}, \quad (5.12)$$

where Φ is the standard univariate normal CDF.

Finally, we collect the transformed vectors for all $t \in T_{train}$ into a matrix $\tilde{\mathbf{D}}$, and calculate the covariance matrix in the transformed space as:

$$\text{Cov}(\tilde{\mathbf{D}}) = \mathbb{E} \left[(\tilde{\mathbf{D}} - \mu_{\tilde{\mathbf{D}}})(\tilde{\mathbf{D}} - \mu_{\tilde{\mathbf{D}}})^T \right], \quad (5.13)$$

where $\mu_{\tilde{\mathbf{D}}}$ is given by the column-wise expectation of the matrix $\tilde{\mathbf{D}}$. $\text{Cov}(\tilde{\mathbf{D}})$ is then the covariance matrix of the Gaussian copula C_G

5.4.2 TRNDP as a minimum cost flow problem

To solve TRNDP and the frequency determination problem simultaneously, we now rigorously formulate it as a minimum cost flow problem.

In Figure 5.10 on the next page, each OD pair is represented by a node in $\{OD_1, \dots, OD_{|\mathcal{N}|}\}$. They are followed by nodes $\{R_0, R_1, \dots, R_C\}$ which represent the candidate routes; R_0 represents the "walking route", which passengers can take if bus capacity is exceeded, or if walking is quicker than taking the bus. Finally, we have nodes $\{N_{11}, \dots, N_{1K}, \dots, N_{C1}, \dots, N_{CK}\}$, per the number of buses assigned to the candidate routes.

In the minimum cost flow formulation, the flow on the edges represent demand flows. In stage one, when demands flow from an OD node to a route node, the passengers travelling between that OD pair are assigned to the route. In stage two, the total demands flowing to a route node, flows into a single node that indicates the number of buses allocated to that route. For instance, if the demands from R_1 flow into node N_{11} , it means that the route R_1 is allocated one bus.

As in all minimum cost flow problem, we have source nodes and sink nodes where the flows are respectively generated and terminated. In this case, the source nodes are the OD nodes $\{OD_1, \dots, OD_{|\mathcal{N}|}\}$. The only sink node is a dummy node D , which is added to the network graph to force flows through the network.

The edges are weighted by the time savings achieved for each passenger that travels on a particular route, whereas the objective is to maximize the flow of demands from the source nodes to a dummy sink node node D . Thus, this formulation is equivalent to solving the TNDPS to give the maximum total travel time savings.

5.4.2.1 Assumptions

We employ the following assumptions in the minimum cost flow formulation.

- *Passenger demands:* Exogenous demand for each OD pair is either given by the ground truth demands for each hour of the day, or by sampling from the predicted demand distribution.
- *Walking time:* The walking time between any two nodes is given by the Manhattan distance between the two nodes and a fixed walking speed.

Figure 5.10: Minimum cost flow formulation for transit route network design problem.

- *Vehicle travel time:* For all routes, the travel time between any two consecutive bus stops is exogenously given and fixed. It is assumed that the dwell time is insignificant and thus not considered in this work. Nevertheless, a constant dwell time can easily be incorporated into the formulations.
- *Choice behaviour:* It is assumed that passengers choose one of several routes to maximize their expected travel time savings compared with respect to walking time.
- *Candidate route set:* For simplicity, the candidate route set is made up of all possible permutations of the bus stop nodes. While a multitude of candidate route set generation procedures exist in the literature (e.g., (Cipriani et al., 2012b, Kılıç and Gök, 2014)), it is not the main focus of this work. Instead, we focus on showing that given demand predictions and a good supply optimization method, it is possible to combine them with a copula to achieve robust optimization.

Notation	Definition
Sets and Indices:	
$\mathcal{N}$	Set of origin and destination nodes.
o	Origin node.
d	Destination node.
c	Candidate route.
Parameters:	
$\beta_{odc}^{(1)}$	Utility for the transit assignment stage, given by the time savings from taking a ride on candidate route c compared to walking directly from origin o to destination d , for each passenger traveling from origin o to destination d .
$\beta_{ck}^{(2)}$	Utility for the buses allocation stage, given by the negative of the average waiting time for candidate route c , with k buses allocated to the route.
τ_c	Route length of candidate route c .
λ_{od}	Total travel demand from origin o to destination d .
W_{od}	Walk time between origin o and destination d .
B_{odc}	Ride time on candidate route c for passengers traveling from origin o to destination d .
W'_{odc}	Total walk time from origin o to boarding node on candidate route c , and walk time from alighting node on candidate route c to destination d .
f_{ck}	Frequency of candidate route c when allocated k buses.
γ	Capacity of each bus.
ν	Number of routes permitted.
C	Number of candidate routes.
K	Fleet size.
Decision variables:	
$X_{odc}^{(1)}$	Demands flowing from node OD_{od} to node R_c .
$X_{ck}^{(2)}$	Demands flowing from node R_c to node N_{ck} .
$\tilde{X}_{ck}$	Equals to 1 if k buses are allocated to candidate route c .

Table 5.2: Notations used in the minimum cost flow formulation.

5.4.2.2 Linear Program Formulation

Let us now formulate the maximum flow problem (Figure 5.10 on the preceding page) as a linear program, which we can then solve through sampling from the predictive distributions. The linear program uses notations as detailed in Table 5.2.

In the linear program formulation, the utility to be maximized is total time savings of all passengers, namely:

$$\max \sum_{o \in \mathcal{N}_D} \sum_{d \in \mathcal{N}_D} \sum_{1 \leq c \leq C} \sum_{1 \leq k \leq K} \beta_{odc}^{(1)} X_{odc}^{(1)} + \beta_{ck}^{(2)} X_{ck}^{(2)}, \quad (5.14)$$

where $\beta_{odc}^{(1)}$ and $\beta_{ck}^{(2)}$ represent the edge costs for stages one and two respectively.

In stage one, the edge costs are given by the time savings per passenger, namely:

$$\beta_{odc}^{(1)} = W_{od} - B_{odc} - W'_{odc}, \forall o, d \in \mathcal{N}_{\mathcal{D}} \quad 1 \leq c \leq C. \quad (5.15)$$

Note that $\beta_{odc}^{(1)}$ does not take into account for waiting time at bus stops. Instead, wait times are represented through $\beta_{ck}^{(2)}$, the average waiting time for candidate route c when allocated k buses, defined as:

$$\beta_{ck}^{(2)} = \frac{\tau_c}{k}, \forall o, d \in \mathcal{N}_{\mathcal{D}}. \quad (5.16)$$

Next, the edges between all OD nodes and node R_0 are assigned zero cost (i.e. zero time savings):

$$\beta_{od0} = 0. \quad (5.17)$$

The maximization is subject to the following constraints.

$$\sum_{o \in \mathcal{N}_{\mathcal{D}}} \sum_{\substack{d \in \mathcal{N}_{\mathcal{D}} \\ o \neq d}} X_{odc}^{(1)} = \sum_{1 \leq k \leq K} X_{ck}^{(2)} \quad \forall 1 \leq c \leq C, \quad (5.18)$$

$$\sum_{0 \leq c \leq C} X_{odc}^{(1)} = \lambda_{od} \quad \forall o, d \in \mathcal{N}_{\mathcal{D}} \quad o \neq d, \quad (5.19)$$

$$\tilde{X}_{ck} = \begin{cases} 1, & X_{ck}^{(2)} > 0 \\ 0, & X_{ck}^{(2)} = 0 \end{cases} \quad \forall 1 \leq c \leq C \quad 1 \leq k \leq K, \quad (5.20)$$

$$\sum_{1 \leq k \leq K} k \tilde{X}_{ck} \leq K \quad \forall 1 \leq c \leq C, \quad (5.21)$$

$$\sum_{1 \leq c \leq C} \sum_{1 \leq k \leq K} k \tilde{X}_{ck} \leq K, \quad (5.22)$$

$$X_{ck}^{(2)} \leq f_{ck} \gamma \quad \forall 1 \leq c \leq C \quad 1 \leq k \leq K, \quad (5.23)$$

$$\sum_{1 \leq c \leq C} \sum_{1 \leq k \leq K} \tilde{X}_{ck} = \nu. \quad (5.24)$$

Constraint (5.18) is the flow conservation constraint on the nodes in the route selection stage. The total flow out of any given OD node is made equal to the demands on the OD pair with constraint (5.19). In constraint (5.20), the binary variable $\tilde{X}_{ck}$ is defined to determine if the link connecting nodes R_c and N_{ck} is allocated any flow. Constraint (5.21) ensures that the number of buses allocated to each candidate route does not exceed the fleet size, while constraint (5.22) sets the maximum total number of buses allocated to all routes. Constraint (5.23) is the capacity constraint, while (5.24) sets the maximum number of routes allowed.

5.4.2.3 Solution through Sampling

We now describe the solution method for the linear program, for any given time t and prediction model $\mathcal{M}$. Indices t and $\mathcal{M}$ are dropped in the following description for conciseness.

Given predicted demands distributions in the form of quantiles $\left(\hat{y}_{od}^{(q)}\right)_{q \in Q}$, we have a random variable for the travel demand of each OD pair which follows the empirical distributions defined by the predicted quantiles as follows:

$$Y_{od} \sim \mathcal{E}\left(\hat{y}_{od}^{(q)}\right) \quad , \quad q \in Q, \quad (5.25)$$

where $\mathcal{E}$ represents the empirical distribution with Cumulative Distribution Function $F_{\mathcal{E}}$, defined as:

$$F_{\mathcal{E}}(y) = \begin{cases} 0 & , y < 0 \\ \frac{y - \hat{y}^{(q_i)}}{\hat{y}^{(q_{i+1})} - \hat{y}^{(q_i)}} + q_i & , y \in \left(\hat{y}^{(q_i)}, \hat{y}^{(q_{i+1})}\right] \\ 1 & , y > \hat{y}^{(q_{|Q|})} \end{cases} . \quad (5.26)$$

By definition, $\hat{y}^{(0)}$ and $\hat{y}^{(1)}$ are given the values of 0 and $\hat{y}^{(|Q|)} + \hat{y}^{(q_1)}$ respectively. Then by using the Gaussian copula $H = \Phi_{\Sigma}$ constructed in Section 5.4.1, we sample k M -dimensional sample demand vectors $\{\lambda_i | 1 \leq i \leq k\}$ as such:

$$\lambda_i = [Y_1^{-1}\Phi_{\Sigma}(s_{i,1}) \quad \cdots \quad Y_M^{-1}\Phi_{\Sigma}(s_{i,M})] , \quad (5.27)$$

where $\mathbf{s}_i = [s_{i,1} \quad \cdots \quad s_{i,M}] \leftarrow \Phi_{\Sigma}$ is a vector drawn from the multivariate Gaussian distribution with covariance matrix Σ .

For each sample demand vector λ_i , we instantiate a corresponding linear program P_i , such that λ_{od} in constraint (5.19) is given by the sampled demand vector λ_i . We then solve P_i to obtain an optimal solution X_i^* for sample s_i . Finally, we aggregate these solutions to obtain the overall optimal solution X^* , as:

$$X^* = \text{mode}[X_1^*, \dots, X_k^*] . \quad (5.28)$$

5.5 Case Study for Stochastic Optimization

So far, we have defined our general supply optimization method. Let us now apply this method to the case study of autonomous shuttles in the Danish university campus, for which we have developed several demand prediction models in Section 5.3.2. Each of these models uses one of various types of Quantile

Figure 5.11: Nodes in our case study, labeled with indices. Prefix 'D' indicates a demand node, while other nodes are bus stops.

Route Set	Itineraries
<i>RS1</i>	0-2-0
<i>RS2</i>	0-2-0, 0-4-0
<i>RS3</i>	0-2-0, 1-2-1

Table 5.3: Commonly recurring solutions in the case study results. Routes in itineraries are given as lists of bus stops in order of bus visitation.

Regression, and yields a marginal predictive distribution for every OD pair in the case study. Now, we pick several of the best performing models, as per Table 5.1 on page 104, and evaluate each of them by feeding its marginals into the supply optimization method.

In the following Sections, we perform this evaluation for 8-Jan-2018, the first day in the test set, hours 08:00, 09:00, . . . , 18:00. The nodes we use for optimization are described in Figure 5.11, and we set the number of samples at $k = 100$. First, we compare optimization with ground truth observations vs. optimization with predictive distributions. Then, we study the robustness of our method against conventional optimization methods.

The optimal itineraries in this case study are sets of routes, where each route is a sequence of bus stops serviced by a single bus. For brevity, we use the notation in Table 5.3 to refer to commonly recurring solutions in the results. Note, however, that the candidate route set consists not only of the few routes in these solutions, but rather of all possible routes through the 5 bus stops.

Hour	GT	Prediction models					
		LQR_4^{Ind}	HP^{Ind}	DNN_1^{Ind}	$GBoost_3^{Ind}$	DNN_1^{Mul}	$GBoost_1^{Mul}$
8	RS1	RS2 (65) * (27)	RS2 (87) * (12)	RS2 (67) RS3 (28) * (5)	RS2 (70) * (13)	RS2 (74) RS3 (23) * (3)	RS2 (100)
9	RS1	* (98)	* (100)	* (100)	* (100)	* (99)	* (99)
10	RS2	RS1 (100)	RS1 (100)	RS1 (100)	RS1 (99) * (1)	RS1 (100)	RS1 (100)
11	RS1	* (92)	* (97)	* (93)	* (98)	* (95)	* (65)
12	RS1	* (91)	* (93)	* (94)	* (100)	* (97)	* (92)
13	RS1	* (100)	* (100)	* (100)	* (99)	* (100)	* (100)
14	RS1	* (98)	* (98)	* (94)	* (65)	* (94)	* (97)
15	RS1	* (96)	* (99)	* (97)	* (99)	* (98)	* (99)
16	RS1	* (90)	* (92)	* (85)	* (94)	* (88)	* (96)
17	RS2	RS1 (97) * (3)	RS1 (100)	RS1 (93) * (7)	RS1 (97) * (3)	RS1 (91) * (9)	RS1 (100)
18	RS2	* (97)	* (100)	* (97)	* (98)	* (95)	* (92)

Table 5.4: Hourly optimal solutions. Column GT lists true optimal solutions, as obtained from ground truth data. Other columns list optimal solutions as obtained through demand sampling for several best performing prediction models. Solutions are ranked according to their number of occurrences (in parentheses), and each list terminates at the same route as GT, denoted by an asterisk.

5.5.1 Optimization with Ground Truth vs. Predictions

Table 5.4 shows that for all prediction models and all hours except 8:00, optimization with predictive distributions yields the *GT solution*, i.e., the same solution as optimization with ground truth observations. Furthermore, these solutions are obtained with high confidence, i.e., the most frequent solution corresponds to more than 90% of the samples from the predictive distribution. Hour 08:00, however, is more challenging: the confidence of optimization varies across models, and model $GBoost_1^{Mul}$ completely misses the GT solution.

Earlier, Table 5.1 on page 104 showed that the *overall* predictive performance of model HP^{Ind} was considerably worse than $GBoost_1^{Mul}$. However, in Table 5.4, HP^{Ind} often yields better *hourly* optimization performance than $GBoost_1^{Mul}$. This apparent discrepancy is explained by Figure 5.12 on the facing page, which displays the *hourly* predictive performance of each model, measured as hourly Mean Tilted Loss over all OD pairs and $q \in Q$. We see that for this portion of the test set, HP^{Ind} is indeed the model that most often yields the lowest hourly *MTL*.

Moreover, Figure 5.12 on the next page juxtaposes hourly prediction quality against optimization quality, measured in mean time savings over the 100 samples from the predictive distribution. We see that in each hour, the mean time savings of different models are quite close, and HP^{Ind} yields the highest time savings. In hour 15:00, $GBoost_1^{Mul}$ is best for both prediction and optimization. In hours 13:00, ..., 18:00, LQR_4^{Ind} is often the best model for prediction, but

Figure 5.12: Prediction vs. optimization quality for best performing prediction models in 2018-Jan-08 08:00,...,18:00. For each hour and model, the left plot shows Mean Tilted Loss over all OD pairs and $q \in Q$, while the right plot shows average time savings (min) over 100 samples from the corresponding joint predictive distribution. The best value in each hour is framed.

not so for optimization.

5.5.2 Comparison to Conventional Optimization Methods

Now, we compare our optimization framework with two conventional optimization strategies: 1) using median estimates for the parameters, and 2) worst-case optimization (i.e., robust optimization). As discussed in Section 5.2.2.2, both of these methods are commonly used to reduce the full distributions of the parameters into point estimates so that the problems can readily be solved. For each hour t and prediction model $\mathcal{M}$, we thus collapse the predictive distribution

Hour	GT	LQR_4^{Ind}			HP^{Ind}			DNN_1^{Ind}			$GBoost_3^{Ind}$			DNN_1^{Mul}			$GBoost_1^{Mul}$		
		P	M	R	P	M	R	P	M	R	P	M	R	P	M	R	P	M	R
8	RS1	RS2	RS2	RS2	RS2	RS2	RS2	RS2	RS2	RS2	RS2	RS2	RS2	RS2	RS2	RS2	RS2	RS2	RS2
9	RS1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
10	RS2	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1
11	RS1	*	*	*	*	*	*	*	RS2	RS2	*	*	*	*	*	*	*	*	RS2
12	RS1	*	*	*	*	*	*	*	*	RS2	*	*	*	*	*	*	*	*	*
13	RS1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
14	RS1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
15	RS1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
16	RS1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
17	RS2	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1	RS1
18	RS2	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

Table 5.5: Hourly optimal solutions using different optimization strategies. GT = optimization with ground truth observations. P = proposed predictive optimization framework, M = conventional optimization using median point estimates, R = robust optimization (worst-case optimization). Asterisks denote same solution as GT.

to point estimates: 1) the 50% quantile for median estimate, and 2) the 95% quantile for worst-case estimate.

The optimization results for all methods are presented in Table 5.5. We see that our optimization method performs mostly on par with the conventional optimization strategies in terms of GT solutions. Furthermore, our framework outperforms the other methods for models DNN_1^{Ind} and $GBoost_1^{Mul}$. We attribute this gain in performance to the ability of our framework to take in full density estimates for the parameters, as opposed to point estimates in the conventional methods.

5.5.3 Online Supply Optimization

For this case study, it takes only a few seconds to both draw a sample from the joint distribution and solve the corresponding linear program instance. Because our optimization method does so independently for multiple samples, it is straightforward to parallelize this step online. Finding the most frequent solution thereafter takes a few seconds as well even among tens of thousands of solutions. For case studies of larger magnitude, the NP-hard optimization problem at hand can be approximated by reducing the candidate route set through various techniques (e.g., (Cipriani et al., 2012b, Kılıç and Gök, 2014)). This results in a tractable problem, which can again be solved in real time.

5.6 Conclusion

In this study, we present an online framework for optimally adapting supply to demand. On the demand side, the framework yields density estimates of future demand in the form of distributions. On the supply side, the framework uses the demand distributions to compute a solution which is optimal in expectation. This differs from previous approaches, which focus mostly on one of the two sides, or which have either too little or too much robustness built into the optimization method.

The setting we consider in this study is predictive routing for a fleet of vehicles. In this setting, there exists a latent spatio-temporal distribution of demand for using the transit service to commute between Origin-Destination (OD) pairs. Given online information about crowd movements in the serviced area (e.g., via a network of sensors), our framework estimates this demand by predicting the marginal distribution of movement counts for each OD pair.

A key component of our framework is the construction of a Gaussian copula, through which the updated marginal distributions are combined into a joint spatio-temporal demand distribution. This joint distribution allows for supply optimization on a full predictive demand distribution, rather than just point estimates of future demand.

On the supply side, our framework uses a novel, demand-responsive optimization method to select an optimal route and frequency for the transit service. This method uses the aforementioned copula to sample from the joint demand distribution, solve a linear optimization program for each sample, and combine the results into an overall optimal solution. As such, our framework applies generally to any stochastic linear program, where the predictive distribution of parameters can be estimated.

We evaluate our framework through an actual case study of mobility in a university campus, for which we have aggregated counts of movements between buildings, as collected from WiFi records. On the demand side, we build and test several prediction models, each of which yields marginal distributions of future demand by estimating multiple CDF quantiles. On the supply side, we then compute an optimal solution on the output of the best performing prediction models. The results show that our framework often yields a solution equal to the solution obtained a posteriori, i.e., with ground truth observations. We also show that our framework performs better than conventional methods for route optimization, which do not utilize full predictive distributions.

5.6.1 Future Work

For future work, we wish to further explore our predictive optimization framework, as follows.

- On the demand side, we intend to test additional methods for prediction under uncertainty, including Bayesian Inference, and Deep Neural Networks with prediction intervals.
- On the supply side, we intend to test our robust optimization method against other instances of the stochastic Transit Routing Network Design Problem.
- We wish to study the performance of our framework with other forms of copula, e.g., Archimedean copulas.
- We are interested in studying the scalability of our framework as a whole, by applying it to additional case studies with larger data sets.

- We intend to derive probabilistic bounds for the accuracy of our stochastic optimization method as a function of the number of samples from the joint demand distribution. Similar probabilistic analysis can be found in (van Hentenryck and Bent, 2009) for a different class of online stochastic routing problems.

Acknowledgement

This research was conducted as part of the LINC project funded by the Urban Innovative Actions, and has also received funding from the People Programme (Marie Curie Actions) of the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Individual Fellowship H2020-MSCA-IF-2016, ID number 745673.

On the Quality Requirements of Demand Prediction for Dynamic Public Transport

Public Transport (PT) is becoming increasingly responsive to the dynamics of mobility demand and so further relies on proper demand predictions. The question is thus: how accurate need such predictions be for effective operation of demand-responsive PT? This Chapter addresses the question through an experimental case study of PT trips in Metropolitan Copenhagen, Denmark, in which we purposely refrain from using any specific prediction models. Instead, we simulate errors in demand predictions (of any model) by sampling from a variety of distributions, and use the noisy predictions to simulate and optimize demand-responsive PT fleets.

The results suggest that the optimized fleet performance is mainly affected by the presence of infrequently large prediction errors and the skew of the error distribution. In particular, non-Gaussian error distributions with negative skew and no bias can yield better fleet optimization vs. a Gaussian error distribution with the same standard deviation. We also obtain that for the studied error distributions, dynamic routing reduces trip time by at least 23% vs. classic, static routing – an estimated saving of 809,000 €/year in Value of Travel Time.

Within the theme of my thesis, this Chapter takes a “meta” approach to studying the effects of predictive uncertainty on Public Transport performance. It is my second work on interfacing demand prediction with supply optimization in the Public Transport domain. The first work appears in Chapter 5, which uses a copula for sampling, and both works use linear program formulations by Asst. Prof. Yu Jiang and Kelvin Lee for predictive optimization. As before, I am the main author, and the other co-authors are Assoc. Prof. Justin Dauwels (my co-supervisor) and Prof. Francisco C. Pereira (my main supervisor). We are currently revising this work for submission to Transportation Research Part C: Emerging Technologies, mainly by changing the sampling method, so that the errors become spatio-temporally correlated.

6.1 Introduction

Public Transport (PT) has traditionally used static itineraries that remain unchanged for months (Ceder, 2016). However, as autonomous mobility advances

and the vision of Smart Cities takes shape, the day approaches when PT becomes dynamic, so that some itineraries are adapted to real-time transport demand (i.e., are demand-responsive) (Bösch et al., 2018, Horaždovský et al., 2018). Meanwhile also, predictive models of transport demand are increasingly used for both long-term and short-term traffic management (Hashemi and Abdelghany, 2015). Future Public Transport should thus naturally employ predictive models for timely adaptation of service per expected transport demand.

The effective operation of demand-responsive PT thus requires that transport demand be accurately estimated ahead of time. For example, more accurate demand predictions can yield better utilization of PT resources, e.g., so that fewer vehicles are used while travel times are also cut shorter. Conversely, errors in predicted demand might lead to sub-optimal routing, thus resulting in the waste of energy, longer waiting times and profit loss.

The main goal of this work is to study the impact of demand prediction accuracy on subsequent performance of demand-responsive PT. To this end, we use a case study of observed PT trips in Metropolitan Copenhagen, Denmark, which we conduct in two steps. First, we simulate prediction errors through various stochastic perturbations of actual observations. Then, we use the noisy predictions to simulate and optimize a variety of dynamic fleets via a common linear programming formulation. Finally, we analyze the results of both steps and draw conclusions on the effect of prediction accuracy on optimization quality.

For a generalizable experimental design, we do not employ any specific demand prediction models, but rather study the impact of prediction errors once any such model has already been fitted. Consequently, we also need not assume any particular sources of predictive uncertainty, such as nonrecurrent traffic disruptions, big-crowd events, improper modeling choices or inherent stochasticity. In addition, we do not commit to any specific error distribution assumption, but rather study error distributions with a wide range of statistical properties. The PT optimization formulation we use is also similar to many formulations in common practice.

Our work differs from existing studies in several respects. First, studies into demand-responsive PT optimization often deal with development of new prediction models and optimization methods. However, we do not aim to develop new models or methods, but rather analyze the impact of prediction accuracy on PT optimization, regardless of any specific model. Second, studies on uncertainty in modeling often aim to detect, explain and reduce it. Conversely, we aim to neither reduce uncertainty nor investigate its causes, but rather analyze what happens when models deviate from normality (e.g., under nonrecurrent disruptions). Third, contrary to previous studies on simulations of non-Gaussian prediction errors, we focus on their subsequent impact on PT performance.

The remainder of this work is organized as follows. Section 6.2 reviews gaps in current studies on uncertainty in modeling and PT optimization. Section 6.3 details our experiments with simulated error distributions and prediction-based optimization. Section 6.4 then provides the experimental results and their analysis. Finally, Section 6.5 recaps the work by discussing its goal, methodology and main results, and Section 6.6 concludes with a list of key findings and future steps.

6.2 Literature Review

6.2.1 Uncertainty in Modeling and the Normality Assumption

Prediction models are subject to uncertainty, which is generally classified as either aleatoric or epistemic (Beaudrie et al., 2016). Aleatoric uncertainty pertains to inherent variability in the modeled phenomenon, e.g., due to natural randomness or unpredictable changes over time and space. It thus cannot be reduced, but can still be approximated through probability distributions. Conversely, epistemic uncertainty originates from incomplete knowledge about the modeled phenomenon, e.g., due to insufficient data or inaccurate observations, and may thus be reduced given further information.

After its reduction, some uncertainty necessarily remains and yields *residuals*, namely, differences between modeled predictions and actually observed values¹. Consequently, when fitting Linear Regression (LR) models, a *Normality Assumption* is commonly employed, whereby residuals are expected to be identically and Normally distributed (Seber and Lee, 2012). In other modeling contexts, however, residuals can be non-Gaussian (Mak, 2000). It may also be impractical to fit a model with Gaussian residuals, depending on data size and quality (Jackson and White, 2018).

The probability density of residuals can thus vary considerably in standard deviation (dispersion around the mean), skew (asymmetry around the mean) and kurtosis (weight of tails). Multiple methods have been devised for *detecting* deviations from normality, including plots, comparison of moments, and statistical tests (Thode, 2002). However, there are far fewer works on the *impact* of such deviations (Ivković et al., 2020), most of which concentrate on errors in model parameter estimates.

Nelson and Granger (1979) discovered that for economic time series, Autoregressive Moving Average (ARMA) models often yield residuals with markedly

¹Definitions of residuals differ by context and specificity (Gourieroux et al., 1987); we use a general definition that befits the context in this work.

non-Gaussian skew and kurtosis. Davies et al. (1980) expanded on this and concluded that the use of non-Gaussian residuals actually allows for a larger selection of models that better represent time series. In the context of chemical analysis, Wolters and Kateman (1989) used Monte Carlo simulations to quantify errors in Least Squares parameters under small deviations from normality.

More recently, He and Raghunathan (2009) used simulated data to examine sequential regression imputation methods under shifted and scaled non-Gaussian distributions: Uniform, Lognormal and t -distribution. They find that mean performance remains quite robust, despite noticeable instability in regression coefficients. To predict debt and bankruptcy of Serbian companies, Ivković et al. (2020) simulate Exponential and Weibull distributed residuals, study the resulting errors in LR parameter estimates, and devise transformations to reduce these errors. Pernot et al. (2020) study non-Gaussian errors in Quantum Machine Learning models and find that mean measures of prediction error depend significantly on the shape of the error distribution. They also note the need for more research into the impact of error distributions on model reliability in general.

Similarly to the above studies, this work too uses simulated perturbations of real data to study the impact of prediction errors under different distributions. However, none of the previous studies apply directly to the Transport domain, and most of them deal with deviations from normality in linear modeling. In contrast, we do not assume any particular modeling form, hence we also neither presume nor try to mitigate any particular sources of uncertainty. Rather, we examine how crucial the effect of prediction errors is on subsequent PT optimization, once such errors necessarily occur for any model in practical use. Per common modeling practices, we assume only that the predictive model has low bias (preferably zero) and is evaluated through mean error measures (e.g., MAPE and RMSNE, as in Section 6.4).

We also note as that as our methodology involves incremental changes to various experimental parameters, this work relates with studies on Sensitivity Analysis (Saltelli et al., 2004). However, Sensitivity Analysis is concerned with apportioning the uncertainty in the output of a given model to its various inputs, whereas this work is concerned neither with the robustness of any specific model nor with sources of uncertainty.

6.2.2 Public Transport Optimization

Existing works on PT planning and design rely heavily on point estimates of future travel demands, which are mostly obtained through manual data collec-

tion, e.g., via transport surveys. These estimates are thus realizations from a latent (i.e., unknown) transport demand distribution, and so are subject to uncertainty and measurement errors. While these errors are widely acknowledged, their subsequent effects on the performance of the optimized PT network are rarely discussed or quantified. Most works evaluate this performance under the simplifying assumption that demands are fixed, exogenously given and accurate (Asadi Bagloee and Ceder, 2011, Szeto and Jiang, 2014). A few other works attempt to mitigate the errors via robust optimization techniques (An and Lo, 2015b) or chance constraints (Wang et al., 2015), both of which might yield unnecessary operational costs.

More recent works on PT optimization take advantage of advancements in big data and machine learning, e.g., for OD estimation (Krishnakumari et al., 2020, Toole et al., 2015) and real-time fleet management (Hadjidimitriou et al., 2020). Readers are referred to (Iliopoulou and Kepaptsoglou, 2019) for a comprehensive review of the application of big data in Public Transportation. As these advancements gain traction in the PT optimization field (Iliopoulou and Kepaptsoglou, 2019, Wang and Qing-dao-er ji, 2019), it becomes ever more important to study the effects of demand prediction accuracy on optimization quality. Further, these effects should also be evaluated in the context of future PT, which will potentially employ dynamically routed vehicles for better demand-responsiveness (Bösch et al., 2018).

To address these needs, this work uses a case study where routes and frequencies of a dynamic fleet adapt to predicted PT demand in hourly intervals. As common in PT optimization, we formulate this via a Mixed-Integer Linear Program (MILP) for the problem of Transit Network Design and Frequency Setting (TNDFS) with constrained fleet size (An and Lo, 2015b, Szeto and Jiang, 2014). While this formulation is general, the dynamic fleet specifically resembles mobility-on-demand (MoD) services (e.g., (Alonso-Mora et al., 2017)) and similarly relies on predicted transport demand, albeit with less flexible routes and stop locations. However, whereas many MoD services operate on the basis of pre-booked rides (Alonso-Mora et al., 2017, Hyland and Mahmassani, 2018), we do not assume any particular source of demand observations. Accordingly, PT demands in this case study are observed only upon their realization, i.e., only when passengers board or alight.

6.3 Experiments

We now present the data, experiments and results.

Figure 6.1: The 6 most active PT stations in Metropolitan Copenhagen, Denmark.

6.3.1 Data

The data consists of PT trips from 1-Jan-2017 to 21-Dec-2018 in Metropolitan Copenhagen, Denmark. These trips are conducted with electronic travel cards (“Rejsekort”) and so account for approximately 1/3 of all bus and train trips (Rejsekort, 2020). For this case study, we use trips between the 6 most active Public Transport (PT) stations, as depicted in Figure 6.1, thus obtaining $6^2 - 6 = 30$ Origin-Destination (OD) pairs with total 2.15 million trips. This allows us to simulate a closed PT system with urban-scale travel distances, and our findings can then be proportionally expanded to larger PT networks.

We aggregate the data by counting hourly trip starts for each OD pair. Then, we draw at random $N = 100$ hours from 2018-Dec-1 00:00, ..., 2018-Dec-21 23:00. The Ground Truth observations are thus the hourly trip counts for each OD pair in these N hours. Based on the Ground Truth observations, we next simulate noisy predictions of PT demand.

6.3.2 Generation of Noisy Demand Predictions

While trip counts are discrete, we assume that the prediction errors are generated from continuous distributions, for several reasons. First, the observed trip counts provide a discrete view into continuous, latent demand for Public Transport. As such, negative PT demand values are possible and could indicate preference to use other transport modes (e.g., bikes, cars, walking, etc.) over PT. In fact, this discrete view is also censored, as observations are bounded from below at zero. Second, the use of continuous error distributions is in line with the bulk of existing literature on non-Gaussian residuals, as reviewed earlier in Section 6.2.

The error distributions we experiment with are all homoscedastic with standard deviation (SD) σ , for $\sigma = 0.5, 1.0, 2.0, 3.0$ ², as follows.

1. Gaussian, i.e., Normal: $\mathcal{N}(0, \sigma^2)$.
2. Uniform: $\mathcal{U}[0, \sqrt{12}\sigma]$.
3. Exponential ($\mathcal{E}$) with scale σ .
4. Negated Exponential, namely, $-\mathcal{E}$.
5. Weibull ($\mathcal{W}$) with scale 1 and shape that corresponds to SD σ ; reduces to $\mathcal{E}$ for $\sigma = 1$.
6. Negated Weibull, namely, $-\mathcal{W}$.

We shift back each distribution by its expected value to obtain zero mean, as illustrated in Figure 6.2 on the next page. Table 6.1 on the following page compares the error distributions with $\mathcal{N}(0, 1)$, the standard Gaussian, in terms of their 3rd and 4th standardized moments, namely, skew and kurtosis. This Table shows that our experiments cover a range of distribution properties: platykurtic, leptokurtic and mesokurtic – i.e., having kurtosis below, above, or equal to $\mathcal{N}(0, 1)$ – as well as positive, negative and zero skew. The distributions also vary in PDF support, which is either finite ($\mathcal{U}$), semi-infinite ($\mathcal{E}$, $-\mathcal{E}$, $\mathcal{W}$, $-\mathcal{W}$) or infinite ($\mathcal{N}$).

For each OD pair (o, d) , shifted error distribution $\mathcal{D}$ and standard deviation σ , we independently draw N samples, $\delta_{1od}^{\mathcal{D}, \sigma}, \dots, \delta_{Nod}^{\mathcal{D}, \sigma} \sim \mathcal{D}$. Then for all $i = 1 \dots N$, we let g_{iod} denote the corresponding Ground Truth observation, and generate noisy predictions as follows:

$$\rho_{iod}^{\mathcal{D}, \sigma} := g_{iod} + \sigma_g \cdot \delta_{iod}^{\mathcal{D}, \sigma}, \quad (6.1)$$

where σ_g is the sample SD of all ground truth observations. The stochastic errors are thus measured in units of the SD of the observations themselves. Furthermore, every $\rho_{iod}^{\mathcal{D}, \sigma}$ is unbiased, as all error distributions have zero mean.

6.3.3 Fleet Optimization

Having generated noisy predictions of PT demand, we now simulate the demand-responsive PT services with varying no. of buses, bus capacity, and percentage of dynamically routed buses. For capacity, our reference value is $110/3 \approx 33$,

²We have also experimented with a finer grained range of σ and obtained consistent results, which we thus omit for brevity.

Figure 6.2: Probability Density Functions (PDFs) of the error distributions with zero mean. Note the different horizontal scale and support of each distribution. Vertical scales differ too and are omitted, because the density value is inconsequential in this work.

σ		$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}$	$-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$
0.5	Skew	0.00	<i>1.30</i>	<i>0.75</i>	-0.75	<i>1.46</i>	-1.46
	Kurt	0.19	1.80	1.50	1.50	2.35	2.35
1.0	Skew	0.00	<i>10.39</i>	<i>6.00</i>	-6.00	<i>6.00</i>	-6.00
	Kurt	3.00	<i>28.80</i>	<i>24.00</i>	<i>24.00</i>	<i>24.00</i>	<i>24.00</i>
2.0	Skew	0.00	<i>83.14</i>	<i>48.00</i>	-48.00	<i>47.75</i>	-47.75
	Kurt	<i>48.00</i>	<i>460.80</i>	<i>384.00</i>	<i>384.00</i>	<i>624.50</i>	<i>624.50</i>
3.0	Skew	0.00	<i>280.59</i>	<i>162.00</i>	-162.00	<i>183.31</i>	-183.31
	Kurt	<i>243.00</i>	<i>2332.80</i>	<i>1944.00</i>	<i>1944.00</i>	<i>4975.26</i>	<i>4975.26</i>

Table 6.1: Skew and kurtosis of error distributions, marked in comparison to $\mathcal{N}(0, 1)$: bold is below, slanted is above, plain is equal.

as the average bus capacity in Metropolitan Copenhagen is approx. 110, and Rejsekort accounts for approx. 1/3 of all PT trips. We vary capacity around this value as $\gamma = 10, 20, 30, 40$; the results later show that this provides sufficient insight into the effect of capacity variability. For each γ , we then choose a fleet size π_γ as the minimal no. statically routed buses, for which optimization is feasible in each of the 100 busiest hours (i.e., with most trips) before 1-Dec-2018.

For percentage of dynamically routed buses, we use $\alpha = 0\%, 10\%, \dots, 40\%$. In particular, $\alpha = 0\%$ corresponds to a completely statically routed fleet, whose performance is thus independent of predictions and error distribution. The results later show that optimization performance converges towards $\alpha = 40\%$, hence we do not increase α further.

Formulation 6.1 Fleet Optimization

Minimize

$$\sum_{\substack{(o,d) \in Q, r \in R, \\ (s,t) \in E}} c_{st}^{(1)} x_{rst}^{od} + \sum_{\substack{(o,d) \in Q, r \in R, \\ s \in V, k=1..\pi_\gamma}} c_{rk}^{(2)} b_{rks}^{od}, \quad (6.2a)$$

subject to

$$\sum_{k=1..\pi_\gamma} b_{rks}^{od} + \sum_{t:(s,t) \in E} x_{rst}^{od} - \sum_{t:(t,s) \in E} x_{rts}^{od} - a_{rs}^{od} = \begin{cases} \rho_{iod}^{\mathcal{D},\sigma} & s = o \\ -\rho_{iod}^{\mathcal{D},\sigma} & s = d \\ 0 & s \notin \{o, d\} \end{cases} \quad \forall (o, d) \in Q, r \in R, s \in V, \quad (6.2b)$$

$$\sum_{s \in V, (o,d) \in Q} b_{rks}^{od} - M y_{rk} \leq 0 \quad \forall r \in R, k = 1..\pi_\gamma, \quad (6.2c)$$

$$\sum_{(o,d) \in Q} x_{rst}^{od} - \gamma k y_{rk} \leq 0 \quad \forall r \in R, (s, t) \in E, \quad (6.2d)$$

$$\sum_{k=1..\pi_\gamma} y_{rk} \leq 1 \quad \forall r \in R, \quad (6.2e)$$

$$\sum_{r \in R, k=1..\pi_\gamma} k y_{rk} \leq \pi_\gamma, \quad (6.2f)$$

$$y_{rk} \leq 0 \quad \forall k < \pi_\gamma - \lceil \alpha \pi_\gamma \rceil, \quad (6.2g)$$

$$y_{rk} \in \{0, 1\}, \quad x_{rst}^{od}, b_{rks}^{od}, a_{rs}^{od} \geq 0 \quad \forall r \in R, s, t \in V, (o, d) \in Q, k = 1..\pi_\gamma. \quad (6.2h)$$

Finally, we solve a mixed-integer linear programming (MILP) formulation to obtain optimal routes and service frequency. This formulation receives as input a directed graph, where nodes correspond to PT stations, and an edge between two nodes exists if buses can travel between the corresponding stations. Each edge is weighted by the corresponding in-vehicle travel time. The input also includes a set of bi-directional routes R , such that for every combination of 2 or more nodes, R contains the shortest acyclic path through all these nodes. The objective is to minimize passengers' total trip time – i.e., waiting time and in-vehicle travel time – by selecting a route for each bus.

As common in PT optimization, we assume that waiting times and in-vehicle travel times are given exogenously; each route uses the same frequency in both directions; and passengers choose the shortest route from origin to destination. The optimizer first truncates any negative predictions to zero, as negative demand implies no passengers. Note that the predictive error PDFs are not truncated, but rather the predictions derived from them.

For any hour $i = 1 \dots N$, Formulation 6.1 defines the linear program per the no-

Sets	V E Q R	nodes arcs OD pairs routes
Indices	s, t (o, d) r $\bar{r}$ k	nodes OD pair route base route that serves all nodes no. buses allocated to a route
Parameters	γ π_γ $c_{st}^{(1)}$ $c_{rk}^{(2)}$ M	capacity of each bus fleet size vehicle travel time from s to t average waiting time for route r , when r is allocated k buses large number
Decision Variables	x_{rst}^{od} b_{rks}^{od} a_{rs}^{od} y_{rk}	flow of passengers traveling from s to t in route r , as part of a trip from o to d flow of passengers boarding route r at s , as part of a trip from o to d , when r is allocated k buses flow of passengers alighting from route r at s , as part of a trip from o to d binary indicator of allocation of k buses to route r

Table 6.2: Notation for Formulation 6.1.

tation in Table 6.2. The objective and constraints are similar to those commonly used for fleet optimization, as follows. In the objective (6.2a), the first sum is total in-vehicle travel time, and the second sum is total waiting time. Constraint (6.2b) uses the demand predictions to impose flow conservation, such that all passengers who board at an origin also alight at their destination. Constraints (6.2c), (6.2d) and (6.2e) ensure that for any route, the same no. buses applies in all corresponding decision variables. Constraint (6.2f) then guarantees there are as many buses as the given fleet size, while constraint (6.2g) imposes the base route on all statically routed buses. Finally, constraint (6.2h) defines the possible set of values for each decision variable.

6.4 Results

The fleet sizes we obtain per Section 6.3.3 are $\pi_\gamma = 38, 19, 13, 10$ for $\gamma = 10, 20, 30, 40$, respectively. Table 6.3 on the facing page further specifies the no. dynamically and statically routed buses for each γ and α . This percentage is seen to correspond to the PDFs in Figure 6.2 on page 134. For example, as σ increases, $-\mathcal{W}$ yields the fewest negative predictions as its PDF shifts towards the positives.

γ	$\alpha = 0\%$		$\alpha = 10\%$		$\alpha = 20\%$		$\alpha = 30\%$		$\alpha = 40\%$	
	D	S	D	S	D	S	D	S	D	S
10	0	38	4	34	8	30	12	26	16	22
20	0	19	2	17	4	15	6	13	8	11
30	0	13	2	11	3	10	4	9	6	7
40	0	10	1	9	2	8	3	7	4	6

Table 6.3: No. dynamically (D) and statically routed (S) buses.

We evaluate prediction quality through two commonly used, unitless measures of mean error: Mean Absolute Percentage Error (MAPE) and Rooted Mean Squared Normalized Error (RMSNE). These measures are defined as follows:

$$\begin{aligned} \text{MAPE}(\mathcal{D}, \sigma) &:= \frac{1}{N} \sum_{i=1}^N \frac{1}{|Q|} \sum_{(o,d) \in Q} \frac{|\rho_{iod}^{\mathcal{D}, \sigma} - g_{iod}|}{\bar{g}} \\ &= \frac{\sigma_g}{\bar{g}|Q|N} \sum_{i=1}^N \sum_{(o,d) \in Q} |\delta_{iod}^{\mathcal{D}, \sigma}|, \end{aligned} \quad (6.3)$$

$$\begin{aligned} \text{RMSNE}(\mathcal{D}, \sigma) &:= \sqrt{\frac{1}{N} \sum_{i=1}^N \frac{1}{|Q|} \sum_{(o,d) \in Q} \left(\frac{\rho_{iod}^{\mathcal{D}, \sigma} - g_{iod}}{\bar{g}} \right)^2} \\ &= \frac{\sigma_g}{\bar{g}\sqrt{|Q|N}} \sqrt{\sum_{i=1}^N \sum_{(o,d) \in Q} \left(\delta_{iod}^{\mathcal{D}, \sigma} \right)^2}, \end{aligned} \quad (6.4)$$

where $\bar{g}$ is the mean of all ground truth observations. Each right-hand side follows from (6.1) and shows the dependency on sampled errors (where in particular, the fractional coefficient is independent of them).

Figure 6.3 on the following page illustrates the MAPE and RMSNE after the optimizer truncates negative predictions. As expected, both MAPE and RMSNE increase as σ increases. When ranking $\mathcal{D}$ from best to worst, we obtain $-\mathcal{W} \leq \mathcal{W} \leq -\mathcal{E} \leq \mathcal{E} \leq \mathcal{N} \leq \mathcal{U}$ for MAPE vs. $-\mathcal{W} \leq -\mathcal{E} \leq \mathcal{N} \leq \mathcal{U} \leq \mathcal{E} \leq \mathcal{W}$ for RMSNE. Let us next analyze how closely the optimization performance follows any of these rankings.

We evaluate optimization performance through several measures, all of which are based on the objective value (i.e., total trip time). For generalizability, our results are mostly given in relative terms. However, to give a basic sense of scale for this case study, we begin with an absolute measure of trip time (min) per passenger. For this, let

$$T_i(\mathcal{D}, \sigma, \alpha, \gamma) := \frac{F_i(\mathcal{D}, \sigma, \gamma, \alpha)}{P_i}, \quad (6.5)$$

Figure 6.3: MAPE and RMSNE after the optimizer truncates negative predictions. Lower is better.

α	$\gamma = 10$	$\gamma = 20$	$\gamma = 30$	$\gamma = 40$
0%	15.0 (± 6.6)	16.8 (± 7.1)	18.4 (± 7.6)	19.9 (± 8.1)
10%	13.2 (± 5.9)	15.5 (± 6.7)	16.8 (± 7.2)	19.4 (± 8.0)
20%	11.8 (± 5.5)	14.2 (± 6.3)	15.9 (± 7.1)	18.1 (± 7.8)
30%	11.2 (± 5.4)	13.5 (± 6.3)	15.5 (± 7.1)	17.3 (± 7.8)
40%	11.1 (± 5.4)	13.4 (± 6.2)	15.4 (± 7.0)	17.2 (± 7.8)

Table 6.4: Mean ($\pm$ SD) of $T_i(\mathcal{D}, \sigma, \alpha, \gamma)$ in min. Lower is better.

where for all $i = 1 \dots N$, P_i is total no. passengers observed in the i 'th hour, and $F_i(\mathcal{D}, \sigma, \gamma, \alpha)$ is their total trip time using the optimized fleet.

Table 6.4 summarizes $T_i(\mathcal{D}, \sigma, \alpha, \gamma)$ through its mean and SD over all $i, \mathcal{D}, \sigma$. The mean and SD are both seen to decrease as γ decreases and/or α increases. Hence as expected, trip times improve if more buses with lower capacity are used and/or more of the buses are dynamically routed. We also note that the mean and SD do not vary much when further separating by $\mathcal{D}$ and σ , for any fixed α, γ . This is a possible consequence of using the same marginal error distribution for all OD pairs, so that their predicted demands shift similarly, thereby balancing each other out during fleet optimization.

We now proceed to measure how much time per passenger is theoretically lost when optimizing with noisy vs. perfect predictions, as:

$$T_{\text{loss}}(\mathcal{D}, \sigma, \gamma, \alpha) := \frac{1}{N} \sum_{i=1}^N \frac{F_i(\mathcal{D}, \sigma, \gamma, \alpha) - F_i(\mathcal{D}, 0, \gamma, \alpha)}{P_i}. \quad (6.6)$$

Figure 6.4 on the facing page illustrates $T_{\text{loss}}(\mathcal{D}, \sigma, \gamma, \alpha)$ using a grid of plots, where rows are ordered by percentage of dynamic buses (α) and columns are ordered by bus capacity (γ). We see that $T_{\text{loss}}(\mathcal{D}, \sigma, \gamma, \alpha)$ increases when either σ or γ increases, as expected. We also see that for any fixed $\sigma > 0$, a partial order on $\mathcal{D}$ emerges as γ and α increase, so that $-\mathcal{W}$ and $-\mathcal{E}$ are significantly better than all other $\mathcal{D}$, and $-\mathcal{W} < -\mathcal{E} < \mathcal{N}$. These properties hold similarly

Figure 6.4: Average minutes lost with noisy vs. perfect predictions ($\bar{f}_{GT}$ in plot titles). Lower is better. Numeric results are in Table 6.6 on page 144.

for RMSNE but not MAPE, as seen earlier in Figure 6.3.

Interestingly, Figure 6.4 also shows that $T_{\text{loss}}(\mathcal{D}, \sigma, \gamma, \alpha)$ generally increases when α increases, i.e., when more dynamic buses are used for the same fleet size. This holds also when normalizing T_{loss} by the theoretical trip time with perfect predictions, namely:

$$T_{\text{loss}}^{\text{rel}}(\mathcal{D}, \sigma, \gamma, \alpha) := \frac{1}{N} \sum_{i=1}^N \frac{F_i(\mathcal{D}, \sigma, \gamma, \alpha) - F_i(\mathcal{D}, 0, \gamma, \alpha)}{F_i(\mathcal{D}, 0, \gamma, \alpha)}, \quad (6.7)$$

as detailed in Table 6.7 on page 145. Still, T_{loss} and $T_{\text{loss}}^{\text{rel}}$ are purely theoretical measurements, because observations cannot realistically be used before they manifest, and the predictive mean rarely captures them perfectly.

Next, we measure the time saved per passenger when using dynamic buses ($\alpha > 0$) vs. a completely statically routed fleet ($\alpha = 0$), namely:

$$T_{\text{gain}}(\mathcal{D}, \sigma, \gamma, \alpha) := \frac{1}{N} \sum_{i=1}^N \frac{F_i(\mathcal{D}, \sigma, \gamma, 0\%) - F_i(\mathcal{D}, \sigma, \gamma, \alpha)}{P_i}, \quad (6.8)$$

as illustrated in Figure 6.5 on the following page. We see that T_{gain} generally improves as capacity and error SD decrease and the percent of dynamic buses increases. The only exception is $T_{\text{gain}}(\mathcal{D}, \sigma, 20, 10\%) > T_{\text{gain}}(\mathcal{D}, \sigma, 30, 20\%)$, for any $\mathcal{D}$ and σ . This is explained by Table 6.3 on page 137: for $\alpha = 10\%$, both $\gamma = 20, 30$ have the same no. dynamic buses, yet there are more statically routed buses for $\gamma = 20$ than for $\gamma = 30$.

Figure 6.5: Average minutes gained with dynamic vs. completely static routing ($\bar{f}_0$ in plot titles). Higher is better. Numeric results are in Table 6.8 on page 146.

α	$\gamma = 10$	$\gamma = 20$	$\gamma = 30$	$\gamma = 40$
10%	429,000	274,000	293,000	17,000
20%	677,000	515,000	392,000	238,000
30%	783,000	615,000	469,000	354,000
40%	809,000	623,000	488,000	382,000

Table 6.5: Minimum yearly economic gains (€). Higher is better.

We also see in Figure 6.5 that as γ and α increase for any σ , a partial order on $\mathcal{D}$ again emerges, with the same properties detected above for T_{loss} and RMSNE. The best is $T_{\text{gain}}(\mathcal{D}, \sigma, 10, 40\%) = 4 \text{ min}$, for $\sigma \leq 0.5$ and any $\mathcal{D}$. The lowest is $T_{\text{gain}}(\mathcal{D}, 3.0, 40, 10\%) = 0.1 \text{ min}$, for $\mathcal{D} \in \{\mathcal{N}, \mathcal{U}, \mathcal{E}\}$.

We further convert T_{gain} to an economic measure per the Danish Value of Travel Time Savings (VTTS), which has recently been estimated at $\nu = 13.43 \text{ €/h}$ (Rich and Vandet, 2019). For each α and γ , we thus take the smallest T_{gain} over all $\mathcal{D}, \sigma$ and multiply it by both ν (in €/min) and average no. trips per year in the studied OD pairs. Table 6.5 provides the results, rounded to 1000 €/year, where the best yearly gains are at least 809,000 € (for $\alpha = 40\%, \gamma = 10$).

Lastly, we measure the average relative gain when using a dynamic vs. completely static fleet, as:

$$T_{\text{gain}}^{\text{rel}}(\mathcal{D}, \sigma, \gamma, \alpha) := \frac{1}{N} \sum_{i=1}^N \frac{F_i(\mathcal{D}, \sigma, \gamma, 0\%) - F_i(\mathcal{D}, \sigma, \gamma, \alpha)}{F_i(\mathcal{D}, \sigma, \gamma, 0\%)}. \quad (6.9)$$

This is detailed in Table 6.9 on page 147, where in particular, the best values of

Figure 6.6: Minimum $T_{\text{gain}}^{\text{rel}}(\mathcal{D}, \sigma, \gamma, \alpha)$ over all $\mathcal{D}, \sigma$. Higher is better.

$T_{\text{gain}}^{\text{rel}}$ are close to 27.5% while the worst are close to 5%. As a concise reference for choosing operational parameters, Figure 6.6 summarizes $T_{\text{gain}}^{\text{rel}}$ in terms of its minimum over all error distributions. For instance, if operational parameters are chosen conservatively, so that each bus is large ($\gamma = 40$) and only $\alpha = 20\%$ of buses are dynamic, then at least 5% relative gain is achieved. However, if the fleet consists of vehicles with low capacity ($\gamma = 10$), many of which are dynamic ($\alpha = 40\%$), then at least 23% relative gain is achieved.

6.5 Discussion

The main goal of this work is to quantify the effects of demand prediction accuracy on the performance of demand-responsive Public Transport (PT). For this, we have used hourly observations of PT trips as a proxy for transport demand, and conducted simulation experiments in two steps. First, we have simulated the output of demand prediction models by perturbing the observations per various distributions, which cover a wide range of statistical properties. Based on these noisy predictions, we have then used a mixed-integer linear program (MILP) with commonly used objective and constraints, to simulate and optimize PT fleets with varying no. statically and dynamically routed buses.

We have obtained that the differences in error distributions do not account for much variability in trip time per passenger. However, the error distributions differ noticeably in two other measures of time per passenger: 1) time theoretically lost with noisy vs. perfect predictions, 2) time gained with dynamic vs. completely static routing. The worst loss per passenger is 1.3 min, while the best time gain per passenger is 4 min, which is more than 27% in relative terms. In economic terms of Value of Time Savings (VTTS), the best gains in this case study are at least 809,000 €/year.

Also in terms of time gains and losses, we have obtained that the error distributions rank more similarly to the RMSNE than the MAPE of predictions. By (6.3) and (6.4), RMSNE is dominated by exceptionally large prediction errors

due to squaring, unlike MAPE. It thus seems that exceptionally large prediction errors, even if few, can strongly influence the performance of dynamic PT.

Finally, we have seen that when the common Normality Assumption is violated, optimization performance can not only worsen but also improve. E.g., compared to the average gains and losses of the Gaussian distribution ($\mathcal{N}$), the Uniform ($\mathcal{U}$) and Weibull ($\mathcal{W}$) are mostly worse, whereas the Negated Weibull ($-\mathcal{W}$) and Negated Exponential ($-\mathcal{E}$) are mostly better. In conjunction with Table 6.1 on page 134, we find that this corresponds well to skew (rather than kurtosis), as $\mathcal{U}$ and $\mathcal{W}$ have positive skew, while $-\mathcal{W}$ and $-\mathcal{E}$ have negative skew.

6.6 Conclusion

In conclusion, the key findings of this work are as follows.

1. For demand-responsive PT, Normally distributed errors can yield less reliable predictions vs. non-normal, unbiased error distributions that have the same standard deviation but smaller skew.
2. A few exceptionally large prediction errors can strongly influence the temporal and economic gains of dynamic PT vs. static PT.
3. However, these gains still generally increase when using more dynamically routed buses (i.e., higher α) with less capacity (i.e., lower γ), regardless of error distribution.
4. The minimum relative gain is 5% when conservatively choosing $\alpha = 20\%$ and $\gamma = 40$ vs. 23% when more liberally choosing $\alpha = 10\%$ and $\gamma = 10$. Figure 6.6 gives a fuller reference for choosing these operational parameters.

For future work, we plan to extend the current study in several respects, as follows. We plan to construct and perturb a joint distribution on all OD pairs (Peled et al., 2019b), and then compare the subsequent optimization stability vs. the marginal distributions in this work. We also plan to quantify the change in performance when optimization utilizes the full predictive distributions instead of the currently common predictive means (Genevay et al., 2016). We further plan to account for the inherent censorship in the data by incorporating truncated error distributions (Greene, 2006).

Acknowledgement

The data was obtained by kind permission of the Danish Transport, Construction and Housing Authority; Movia Transit Agency; and the Danish Metro Company. This work was supported by the European Union's Horizon 2020 research and innovation programme under Individual Fellowship H2020-MSCA-IF-2016, ID 745673; and the Singaporean Ministry of Education, via the Interdisciplinary Graduate School of Nanyang Technological University.

6.7 Supplementary Results

Additional numeric results appear in the following Tables.

α	σ	$\gamma = 10$			$\gamma = 20$			$\gamma = 30$			$\gamma = 40$		
		$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}$	$-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$	$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}$	$-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$
10%	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.1	0.1	0.1	0.1	0.1
	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.2	0.2	0.1	0.2	0.1
20%	3.0	0.1	0.1	0.1	0.1	0.1	0.1	0.3	0.4	0.5	0.3	0.3	0.2
	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.0
	1.0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.1	0.2	0.1
30%	2.0	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.4	0.4	0.2	0.4	0.3
	3.0	0.4	0.4	0.3	0.3	0.3	0.2	0.6	0.7	0.9	0.3	0.6	0.7
40%	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.0
	1.0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.1	0.2	0.1
	2.0	0.3	0.2	0.3	0.1	0.3	0.1	0.2	0.4	0.5	0.2	0.5	0.3
40%	3.0	0.5	0.5	0.7	0.2	0.6	0.1	0.5	0.6	0.8	1.0	0.9	1.1
	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.1	0.1	0.1	0.1	0.1
	1.0	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.3	0.3	0.2	0.3	0.2
40%	2.0	0.3	0.2	0.3	0.1	0.3	0.1	0.4	0.6	0.7	0.4	0.7	0.5
	3.0	0.5	0.5	0.7	0.2	0.6	0.1	0.7	0.7	0.9	0.5	1.0	0.9

Table 6.6: $T_{\text{loss}}(\mathcal{D}, \sigma, \gamma, \alpha)$ in min, lower is better.

α	σ	$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}$	$\gamma=10$ $-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$	$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}$	$\gamma=20$ $-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$	$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}$	$\gamma=30$ $-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$	$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}$	$\gamma=40$ $-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$
10%	0.5	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.2	0.1	0.2	0.2	0.1	0.2	0.2	0.1	0.2	0.1
	1.0	0.1	0.2	0.2	0.3	0.2	0.1	0.3	0.3	0.2	0.2	0.2	0.3	0.4	0.3	0.4	0.7	0.5	0.4	0.3	0.3	0.4	0.3	0.4	0.2
	2.0	0.3	0.3	0.2	0.3	0.3	0.3	0.5	0.7	0.4	0.5	0.4	0.5	0.7	1.4	1.2	0.9	1.1	0.7	1.4	1.5	1.6	0.5	1.3	0.5
	3.0	0.7	0.6	0.8	0.5	0.7	0.2	0.8	0.7	1.1	0.9	0.7	0.5	1.8	1.9	2.6	1.6	2.1	0.6	2.5	2.8	2.4	2.0	1.8	1.1
20%	0.5	0.3	0.3	0.3	0.3	0.3	0.4	0.3	0.3	0.2	0.3	0.3	0.2	0.2	0.1	0.1	0.2	0.1	0.2	0.3	0.2	0.2	0.2	0.2	0.2
	1.0	1.0	1.1	1.1	0.9	1.1	0.8	0.6	0.9	1.1	0.5	1.1	0.5	0.3	0.7	1.0	0.4	1.0	0.4	0.5	0.8	0.8	0.4	0.8	0.4
	2.0	2.0	2.1	2.0	1.8	1.8	1.8	1.4	1.9	1.7	1.2	1.5	1.2	1.5	2.6	2.3	1.4	2.3	1.4	2.3	3.0	2.9	1.4	2.4	1.5
	3.0	3.3	3.7	3.7	2.6	2.2	2.4	3.0	2.7	3.4	1.9	2.2	1.6	3.6	4.6	5.5	2.0	3.8	1.7	4.6	5.0	5.1	3.4	4.0	1.8
30%	0.5	0.2	0.1	0.2	0.1	0.1	0.1	0.2	0.3	0.3	0.2	0.3	0.3	0.3	0.2	0.2	0.1	0.2	0.1	0.2	0.2	0.2	0.1	0.2	0.1
	1.0	0.4	0.5	0.6	0.4	0.6	0.2	0.6	0.8	0.7	0.6	0.7	0.4	0.5	0.8	0.8	0.5	0.8	0.5	0.6	1.0	0.8	0.5	0.8	0.5
	2.0	2.3	2.3	2.5	1.0	2.4	0.7	1.9	2.5	2.5	1.3	2.2	1.0	1.5	2.8	3.0	1.3	2.9	1.4	3.0	3.7	3.9	1.6	3.3	1.6
	3.0	4.2	4.8	5.5	1.8	4.9	1.1	3.8	4.3	5.5	2.2	4.8	1.9	4.1	5.0	6.5	2.4	5.4	2.0	5.9	7.1	7.1	4.1	6.2	2.7
40%	0.5	0.2	0.2	0.3	0.2	0.3	0.2	0.5	0.5	0.6	0.3	0.6	0.4	0.5	0.7	0.7	0.5	0.6	0.6	0.6	0.6	0.6	0.4	0.6	0.4
	1.0	0.6	0.7	0.9	0.5	0.9	0.4	1.1	1.1	1.4	0.9	1.4	0.8	1.3	1.5	1.8	1.1	1.8	1.1	1.2	1.6	1.7	1.1	1.7	1.1
	2.0	2.5	2.2	3.2	0.9	3.2	0.7	2.6	3.2	3.4	1.7	3.4	1.4	2.6	3.8	4.3	2.0	4.5	2.0	3.6	4.4	5.4	2.3	4.3	2.3
	3.0	4.8	4.3	6.0	2.1	5.8	1.0	4.9	5.4	6.5	3.6	6.0	2.8	5.6	6.0	7.2	3.5	6.7	2.9	7.1	7.5	7.1	5.0	6.8	3.4

Table 6.7: $T_{\text{loss}}^{\text{rel}}(\mathcal{D}, \sigma, \gamma, \alpha)$ in %, lower is better.

α	σ	$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}^{\gamma=10}$	$-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$	$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}^{\gamma=20}$	$-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$	$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}^{\gamma=30}$	$-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$	$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}^{\gamma=40}$	$-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$
10%	0.0	1.9	1.9	1.9	1.9	1.9	1.9	1.3	1.3	1.3	1.3	1.3	1.3	1.7	1.7	1.7	1.7	1.7	1.7	1.7	0.6	0.6	0.6	0.6	0.6
	0.5	1.9	1.9	1.9	1.9	1.9	1.9	1.3	1.3	1.3	1.3	1.3	1.3	1.6	1.6	1.6	1.6	1.6	1.6	1.6	0.5	0.5	0.5	0.5	0.6
	1.0	1.9	1.9	1.9	1.9	1.9	1.9	1.3	1.3	1.3	1.3	1.3	1.3	1.6	1.6	1.6	1.6	1.6	1.6	1.6	0.5	0.5	0.5	0.5	0.5
	2.0	1.9	1.9	1.9	1.9	1.9	1.8	1.2	1.2	1.3	1.2	1.3	1.2	1.5	1.5	1.5	1.5	1.5	1.5	1.5	0.3	0.3	0.3	0.3	0.5
20%	3.0	1.8	1.8	1.8	1.8	1.8	1.9	1.2	1.2	1.1	1.2	1.2	1.2	1.4	1.3	1.2	1.4	1.3	1.6	1.6	0.1	0.1	0.1	0.2	0.4
	0.0	3.3	3.3	3.3	3.3	3.3	3.3	2.7	2.7	2.7	2.7	2.7	2.7	2.5	2.5	2.5	2.5	2.5	2.5	2.5	1.9	1.9	1.9	1.9	1.9
	0.5	3.2	3.2	3.2	3.2	3.2	3.2	2.6	2.6	2.6	2.6	2.6	2.6	2.5	2.5	2.5	2.5	2.5	2.5	2.5	1.8	1.8	1.8	1.8	1.8
	1.0	3.2	3.1	3.1	3.1	3.1	3.1	2.6	2.5	2.5	2.5	2.5	2.5	2.5	2.4	2.3	2.4	2.3	2.4	2.3	1.8	1.7	1.7	1.7	1.8
30%	2.0	3.0	3.0	3.1	3.1	3.1	3.1	2.4	2.4	2.4	2.4	2.5	2.5	2.3	2.1	2.1	2.3	2.1	2.3	2.1	1.5	1.4	1.4	1.6	1.5
	3.0	2.9	2.8	2.8	3.0	3.0	3.0	2.2	2.3	2.1	2.4	2.3	2.3	1.9	1.8	1.6	2.2	1.9	2.3	2.3	1.1	1.0	1.0	1.3	1.2
	0.0	3.9	3.9	3.9	3.9	3.9	3.9	3.4	3.4	3.4	3.4	3.4	3.4	3.0	3.0	3.0	3.0	3.0	3.0	3.0	2.7	2.7	2.7	2.7	2.7
	0.5	3.9	3.9	3.9	3.9	3.9	3.9	3.3	3.3	3.3	3.3	3.3	3.3	2.9	2.9	2.9	2.9	2.9	2.9	2.9	2.6	2.6	2.6	2.6	2.7
40%	1.0	3.9	3.9	3.8	3.9	3.8	3.9	3.3	3.3	3.2	3.3	3.2	3.2	2.9	2.9	2.8	2.9	2.8	2.9	2.8	2.6	2.5	2.5	2.5	2.6
	2.0	3.7	3.7	3.6	3.8	3.6	3.8	3.1	3.1	3.0	3.2	3.1	3.1	2.7	2.6	2.5	2.8	2.5	2.7	2.7	2.2	2.1	2.0	2.4	2.1
	3.0	3.5	3.4	3.2	3.7	3.3	3.8	2.8	2.7	2.6	3.1	2.7	3.1	2.4	2.2	1.9	2.6	2.1	2.7	2.7	1.8	1.5	1.5	2.0	1.6
	0.0	4.0	4.0	4.0	4.0	4.0	4.0	3.5	3.5	3.5	3.5	3.5	3.5	3.2	3.2	3.2	3.2	3.2	3.2	3.2	2.8	2.8	2.8	2.8	2.8
	0.5	4.0	4.0	4.0	4.0	4.0	4.0	3.4	3.4	3.4	3.4	3.4	3.4	3.1	3.0	3.0	3.1	3.0	3.1	3.0	2.7	2.7	2.7	2.7	2.8
	1.0	3.9	3.9	3.9	3.9	3.9	4.0	3.3	3.3	3.3	3.3	3.3	3.4	2.9	2.9	2.8	3.0	2.8	2.9	2.9	2.6	2.5	2.5	2.6	2.6
	2.0	3.7	3.8	3.7	3.9	3.7	3.9	3.1	3.1	3.0	3.3	3.0	3.3	2.7	2.6	2.5	2.8	2.4	2.8	2.7	2.2	2.1	1.9	2.4	2.1
	3.0	3.5	3.5	3.4	3.8	3.4	3.9	2.8	2.7	2.6	3.0	2.7	3.1	2.3	2.2	2.0	2.6	2.1	2.7	2.7	1.7	1.6	1.6	2.0	1.7

Table 6.8: $T_{\text{gain}}(\mathcal{D}, \sigma, \gamma, \alpha)$ in min, higher is better.

α	σ	$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}$	$-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$	$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}$	$-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$	$\mathcal{N}$	$\mathcal{U}$	$\mathcal{E}$	$-\mathcal{E}$	$\mathcal{W}$	$-\mathcal{W}$
10%	0.0	12.7	12.7	12.7	12.7	12.7	12.7	8.2	8.2	8.2	8.2	8.2	8.2	9.5	9.5	9.5	9.5	9.5	9.5
	0.5	12.6	12.6	12.6	12.7	12.7	12.6	8.0	8.1	8.1	8.1	8.1	8.0	9.3	9.3	9.3	9.3	9.3	9.3
	1.0	12.6	12.5	12.6	12.5	12.6	12.7	7.9	7.9	8.0	7.9	8.0	7.9	9.1	9.0	9.0	8.9	9.0	9.1
	2.0	12.5	12.5	12.5	12.5	12.5	12.5	7.7	7.5	7.8	7.7	7.8	7.7	8.8	8.2	8.3	8.7	8.5	8.9
	3.0	12.1	12.2	12.1	12.3	12.2	12.5	7.4	7.5	7.2	7.3	7.6	7.7	7.8	7.8	7.1	8.0	7.6	8.9
20%	0.0	22.2	22.2	22.2	22.2	22.2	22.2	16.3	16.3	16.3	16.3	16.3	16.3	14.3	14.3	14.3	14.3	14.3	14.3
	0.5	22.0	21.9	22.0	22.0	21.9	21.9	16.1	16.1	16.1	16.1	16.1	16.1	14.1	14.2	14.2	14.1	14.2	14.1
	1.0	21.5	21.3	21.4	21.5	21.4	21.6	15.8	15.6	15.4	15.9	15.4	15.9	14.1	13.7	13.5	14.0	13.5	13.9
	2.0	20.7	20.6	20.7	20.9	20.8	20.9	15.2	14.8	14.9	15.3	15.1	15.3	13.1	12.1	12.3	13.1	12.4	13.2
	3.0	19.7	19.3	19.4	20.2	20.5	20.3	13.9	14.1	13.5	14.7	14.4	15.0	11.3	10.4	9.6	12.6	11.0	12.9
30%	0.0	26.8	26.8	26.8	26.8	26.8	26.8	20.8	20.8	20.8	20.8	20.8	20.8	17.1	17.1	17.1	17.1	17.1	17.1
	0.5	26.7	26.7	26.7	26.7	26.7	26.7	20.6	20.6	20.5	20.6	20.5	20.5	16.8	16.9	16.9	16.9	16.9	16.9
	1.0	26.5	26.5	26.4	26.5	26.4	26.6	20.3	20.2	20.2	20.3	20.2	20.3	16.6	16.4	16.3	16.6	16.3	16.6
	2.0	25.2	25.2	25.0	26.1	25.1	26.3	19.3	18.8	18.8	19.7	19.0	20.0	15.8	14.8	14.6	16.0	14.7	15.9
	3.0	23.7	23.3	22.8	25.5	23.2	26.0	17.8	17.4	16.4	19.0	17.0	19.2	13.7	12.9	11.6	15.1	12.6	15.4
40%	0.0	27.5	27.5	27.5	27.5	27.5	27.5	21.5	21.5	21.5	21.5	21.5	21.5	17.9	17.9	17.9	17.9	17.9	17.9
	0.5	27.3	27.3	27.3	27.4	27.3	27.4	21.1	21.1	21.0	21.2	21.0	21.2	17.4	17.3	17.3	17.4	17.3	17.4
	1.0	27.1	27.1	26.9	27.1	26.9	27.2	20.6	20.7	20.4	20.8	20.4	20.9	16.8	16.7	16.4	17.0	16.4	16.9
	2.0	25.7	26.0	25.3	26.9	25.2	27.0	19.5	19.0	18.8	20.2	18.9	20.4	15.7	14.7	14.3	16.2	14.2	16.2
	3.0	24.0	24.4	23.2	26.0	23.3	26.8	17.7	17.3	16.4	18.7	16.8	19.4	13.3	12.9	11.9	15.0	12.3	15.5

Table 6.9: $T_{\text{gain}}^{\text{rel}}(\mathcal{D}, \sigma, \gamma, \alpha)$ in %, higher is better.

Conclusion and Future Work

In this thesis, we have studied several Machine Learning models of Transportation under uncertainty, while also accounting for interdependence between Transportation demand and supply. For uncertainty under nonrecurrent traffic disruptions, such as road incidents, we have presented in Chapter 2 QTIP: a real-time simulation-based framework for quick model adaptation. For uncertainty in actual mobility demand, as observed through limited shared supply, we have suggested two non-parametric censored methods in Chapters 3 and 4. For dynamic operation of demand-responsive mobility services, we have studied the effects of demand uncertainty via two case studies: on-campus autonomous mobility in Chapter 5, and dynamic bus fleets in a metropolitan area in Chapter 6. We next consolidate our findings and key messages and then describe our plans for future work.

7.1 Findings and Key Messages

On one hand, we have demonstrated that predictive quality can deteriorate if some factors that affect the dynamics of mobility demand are insufficiently accounted for. Chapter 2 demonstrates such deterioration in models that are not adapted to the specific characteristics of each road incident. Chapters 3 and 4 illustrate the degradation in predictive quality when insufficiently accounting for inherent censorship. Chapter 5 shows that not using a full predictive distribution can lead to sub-optimal performance of demand-responsive mobility services. For such services too, Chapter 6 shows that predictions with Gaussian errors can be less reliable than predictions with errors that are non-Gaussian, unbiased and negatively skewed.

On the other hand, we have shown that this deterioration can be mitigated through several proposed methods of demand modeling. By leveraging distress signals from vehicles involved in incidents, QTIP obtains on average 28.74% Rooted Mean Squared Error (RMSE) improvement vs. non-adapted models in the Danish highway case study in Chapter 2. The censored case studies on bike-sharing and taxi data in Chapter 3 show that incorporating censorship in the likelihood function of Gaussian Processes can improve RMSE and Coefficient of Determination (R^2) vs. ignoring the censorship or treating it via omission or imputation. As an alternative method, Chapter 4 shows that Censored Quantile Regression Neural Networks too can improve predictive quality

vs. both censorship-unaware models and parametric models, as demonstrated on bike-sharing and shared Electric Vehicles data. The framework for predictive optimization of Public Transport in Chapter 5 estimates a full distribution of latent demand and so optimizes routes better than conventional techniques that do not exploit the full predictive distribution. The case study in Chapter 6 shows that for a wide variety of error distributions, dynamically routed Public Transport can significantly reduce travel times when the standard deviation of prediction errors is at most 0.5.

In fact, the advantages of retaining a full uncertainty structure of future demand recur throughout this thesis. Chapters 4 and 5 advocate to do so with Quantile Regression and study its benefits through several regression forms, including Linear, Gradient Boosted and Neural Network-based. Appendix A presents both Probabilistic Graphical Models for fully specifying a stochastic generative process and Bayesian Inference for updating the corresponding distribution per data (as also used in Chapter 2). Multiple case studies in Appendix A also demonstrate the power and convenience of modeling with these probabilistic methods.

Our works also show that for some Transportation modeling tasks, higher complexity and newer methods are not necessarily better than relative simplicity and classic methods. For instance, Linear Regression models outperform Neural Networks and Gradient Boosting in the incidents case study in Chapter 2. Also, for Quantile Regression of real-world mobility data in Chapter 4, shallow Neural Networks with linear activations outperform Deep Neural Networks with nonlinear activations (and regularization). In some cases though, higher modeling complexity does improve predictive quality, e.g., by adding memory to Neural Networks (Chapter 4) or using multivariate modeling (Chapter 5). Appendix B further demonstrates the benefits of these two enhancements in a case study of modeling traffic in Copenhagen.

In conclusion, our findings carry several encouraging messages to practitioners and theoreticians of current and prospective Transportation, as follows.

1. Our findings suggest that recent technological advances in vehicular systems offer a feasible opportunity for mitigating the long-standing problem of instantaneous model adaptation upon road incidents. In particular, these findings hold in circumstances of high traffic demand, during which efficient traffic management is most needed.
2. Our findings emphasize that the inherent censorship in Transport demand should be explicitly modeled to alleviate bias, and we provide non-parametric models that can successfully do so even under severe censorship.
3. Our findings suggest that demand-responsive (and autonomous) Transport

services can be routed more optimally when retaining a full uncertainty structure of mobility demand. For that matter too, we also find that deviations from the common Normality Assumption can in fact improve service performance.

4. Our findings reaffirm that in any modeling task, relatively simple and traditional models must be considered not only for baseline comparison to newer models, but also as viable alternatives to unnecessarily complex models.

7.2 Future Work

We plan to extend our works in several directions, as follows.

- We wish to further study the capabilities and computational scalability of our frameworks.
 - For QTIP, we plan to experiment with additional incident scenarios that involve larger networks, other road topology and different disruptions, as well as varying quality of distress signals. We further plan to study QTIP in a meta-learning setting, wherein it evaluates several parametric and non-parametric model forms and ranks them on a per-incident basis.
 - For the Public Transport predictive optimization framework, we also wish to study larger networks with more stops and greater variability of service frequency.
- We also plan to enhance our models through additional Machine Learning methods, which our works have not utilized comprehensively. To this end, we plan to more often use Probabilistic Graphical Models and Bayesian Inference when modeling uncertainty in traffic and mobility demand. When modeling with Censored Neural Networks, we plan to smooth the loss function with differentiable approximations for possibly better training with backpropagation. For modeling additional time series data, we plan to increase our use of ARIMA, Gradient Boosting and other popular methods.
- In particular, we plan to make more use of multi-task learning methods, which have gained traction in recent years (Zhang and Yang, 2018). When modeling mobility demand, we plan to further leverage spatio-temporal correlations through multivariate (censored) modeling (Chu et al., 2019, Petersen et al., 2019, Toman et al., 2020). In non-parametric modeling with Quantile Regression, we shall jointly model quantiles using Deep Neural Networks, as in (Rodrigues and Pereira, 2020).

- We also plan to integrate our works in several real-world Transportation systems. In year 2021, the predictive optimization framework will undergo integration with an autonomous shuttle service in DTU Lyngby campus (LINC, 2020). Beginning next year too, we will extend our methods of censored regression and latent distribution estimation to a long-term project (named “BOON”) for optimal planning of Electric Vehicle charging stations in Denmark. We also consider piloting QTIP as a component in a real-time traffic management system, such as PtMS (Kong et al., 2013) or DynaMIT2.0 (Lu et al., 2015), possibly with online calibration (Antoniou et al., 2007, Prakash et al., 2018, Qin and Mahmassani, 2004).

References

- Abdullah, Z. (2018). Ntu gets new driverless shuttle bus to ferry students across campus. <https://www.straitstimes.com/singapore/transport/ntu-gets-new-driverless-shuttle-bus-to-ferry-students-across-campus>.
- Abramowitz, M. and Stegun, I. A., editors (1965). *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables, sec 9.6*. Dover Publications, Inc., New York.
- Aghabayk, K., Sarvi, M., Young, W., and Kautzsch, L. (2013). A novel methodology for evolutionary calibration of vissim by multi-threading. In *36th Australasian Transp. Res. Forum*, pages 1–15.
- Albiński, S., Fontaine, P., and Minner, S. (2018). Performance analysis of a hybrid bike sharing system: A service-level-based approach under censored demand observations. *Transportation research part E: logistics and transportation review*, 116:59–69.
- Aldrich, J. H., Nelson, F. D., and Adler, E. S. (1984). The linear probability model. In *Linear probability, logit, and probit models*, volume 45, chapter 1, pages 9–27. Sage.
- Allik, B., Miller, C., Piovosio, M. J., and Zurakowski, R. (2015). The tobit kalman filter: an estimator for censored measurements. *IEEE Transactions on Control Systems Technology*, 24(1):365–371.
- Alonso-Mora, J., Wallar, A., and Rus, D. (2017). Predictive routing for autonomous mobility-on-demand systems with ride-sharing. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 3583–3590. IEEE.
- Alton, L. (2018). Driverless cars will impact healthcare for better and worse. <https://www.healthworkcollective.com/driverless-cars-will-impact-healthcare-better-worse/>.
- Amemiya, T. (1984). Tobit models: A survey. *Journal of econometrics*, 24(1-2):3–61.
- Amemiya, T. (1985). Tobit models. In *Advanced Econometrics*, page 387. Harvard university press.
- Amiripour S. M. Mahdi, Ceder Avishai (Avi), and Mohaymany Afshin Shariat (2014). Hybrid method for bus network design with high seasonal demand variation. *J. Transp. Eng.*, 140(6):04014015.
- An, K. and Lo, H. K. (2015a). Robust transit network design with stochastic demand considering development density. *Transportation Research Part B: Methodological*, 81:737–754.
- An, K. and Lo, H. K. (2015b). Robust transit network design with stochastic demand considering development density. *Transportation Research Part B: Methodological*, 81:737 – 754.
- An, K. and Lo, H. K. (2016). Two-phase stochastic program for transit network design under demand uncertainty. *Transportation Research Part B: Methodological*, 84:157–181.
- Antoniou, C., Barceló, J., Breen, M., Bullejos, M., Casas, J., Cipriani, E., Ciuffo, B., Djukic, T., Hoogendoorn, S., Marzano, V., Montero, L., Nigro, M., Perarnau, J., Punzo, V., Toledo, T., and van Lint, H. (2016). Towards a generic benchmarking platform for origin–destination flows estimation/updating algorithms: Design, demonstration and validation. *Transportation Research Part C: Emerging Technologies*, 66:79–98.
- Antoniou, C., Ben-Akiva, M., and Koutsopoulos, H. N. (2007). Nonlinear kalman filtering algorithms for on-line calibration of dynamic traffic assignment models. *IEEE Trans. Intell. Transp. Syst.*, 8(4):661–670.
- Antoniou, C., Dimitriou, L., and Pereira, F. (2018). *Mobility patterns, big data and transport analytics: tools and applications for modeling*. Elsevier.
- Antunes, F., O’Sullivan, A., Rodrigues, F., and Pereira, F. (2017). A review of heteroscedasticity treatment with gaussian processes and quantile regression meta-models. In *Seeing Cities Through Big Data*, pages 141–160. Springer.
- Asadi Bagloee, S. and Ceder, A. (2011). Transit-network design methodology for actual-size road networks. *Transportation Research Part B: Methodological*, 45(10):1787 – 1804.

- Athey, S. (2018). The impact of machine learning on economics. In *The economics of artificial intelligence: An agenda*, pages 507–547. University of Chicago Press.
- Balakrishna, R., Antoniou, C., Ben-Akiva, M., Koutsopoulos, H. N., and Wen, Y. (2007). Calibration of microscopic traffic simulation models: Methods and application. *Transportation Research Record*, 1999(1):198–207.
- Barua, L., Zou, B., and Zhou, Y. (2020). Machine learning for international freight transportation management: a comprehensive review. *Research in Transportation Business & Management*, 34:100453.
- Baykal-Gursoy, M., Xiao, W., Duan, Z., and Ozbay, K. (2006). Delay estimation for traffic flow interrupted by incidents. In *86th Transp. Res. Board Annu. Meet.*
- Beaudrie, C. E., Kandlikar, M., and Ramachandran, G. (2016). *Using Expert Judgment for Risk Assessment*. Elsevier Inc., 2 edition.
- Ben-Akiva, M., Bierlaire, M., Koutsopoulos, H. N., and Mishalani, R. (1998). Dynamit: a simulation-based system for traffic prediction. In *DACCORD Short Term Forecast. Workshop*, pages 1–12.
- Ben-Akiva, M., Bierlaire, M., Koutsopoulos, H. N., and Mishalani, R. (2002). Real time simulation of traffic demand-supply interactions within dynamit. In *Transportation and network analysis: current trends*, pages 19–36. Springer.
- Ben-Akiva, M., Koutsopoulos, H. N., Antoniou, C., and Balakrishna, R. (2010). Traffic simulation with dynamit. In Barceló, J., editor, *Fundamentals of Traffic Simulation*, pages 363–398. Springer New York, New York, NY.
- Ben-Akiva, M., Koutsopoulos, H. N., and Mukundan, A. (1994). A dynamic traffic model system for atms/atis operations. *J. Intell. Transp. Syst.*, 2(1):1–19.
- Bernardo, J. M. and Smith, A. F. (2009). *Bayesian theory*, volume 405. John Wiley & Sons.
- Bernas, M., Placzek, B., Korski, W., Loska, P., Smyła, J., and Szymala, P. (2018). A survey and comparison of low-cost sensing technologies for road traffic monitoring. *Sensors*, 18(10):3243.
- Bertini, R. L., Rose, M. W., and El-Geneidy, A. (2005). Using archived its data sources to measure the effectiveness of a freeway incident response program. In *84th Transp. Res. Board Annu. Meet.*, pages 9–13.
- Bewick, V., Cheek, L., and Ball, J. (2004). Statistics review 12: survival analysis. *Critical care (London, England)*, 8(5):389–394.
- Bhavsar, P., Safo, I., Bouaynaya, N., Polikar, R., and Dera, D. (2017). Machine learning in transportation data analytics. In Chowdhury, M., Apon, A., and Dey, K., editors, *Data Analytics for Intelligent Transportation Systems*, pages 283 – 307. Elsevier.
- Biganzoli, E., Boracchi, P., and Marubini, E. (2002). A general framework for neural network models on censored survival data. *Neural Networks*, 15(2):209–218.
- Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer-Verlag, Berlin, Heidelberg.
- Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*.
- Bösch, P. M., Becker, F., Becker, H., and Axhausen, K. W. (2018). Cost-based analysis of autonomous mobility services. *Transport Policy*, 64:76–91.
- Boyce, D., Lee, D.-H., and Ran, B. (2001). Analytical models of the dynamic traffic assignment problem. *Netw. Spat. Econ.*, 1(3-4):377–390.
- Branda, F., Marozzo, F., and Talia, D. (2020). Ticket sales prediction and dynamic pricing strategies in public transport. *Big Data and Cognitive Computing*, 4(4):36.

- Brandl, O. (2016). V2x traffic management. *e & i Elektrotechnik und Informationstechnik*, 133(7):353–355.
- Brownlee, J. (2017). On the suitability of long short-term memory networks for time series forecasting. <https://machinelearningmastery.com/suitability-long-short-term-memory-networks-time-series-forecasting/>.
- Buchinsky, M. (1998). Recent advances in quantile regression models: a practical guideline for empirical research. *Journal of human resources*, pages 88–126.
- Cannon, A. J. (2011). Quantile regression neural networks: Implementation in r and application to precipitation downscaling. *Computers & geosciences*, 37(9):1277–1284.
- Carrier, G., Chernozhukov, V., and Galichon, A. (2016). Vector quantile regression: an optimal transport approach. *The Annals of Statistics*, 44(3):1165–1192.
- Castro-Neto, M., Jeong, Y.-S., Jeong, M.-K., and Han, L. D. (2009). Online-svr for short-term traffic flow prediction under typical and atypical traffic conditions. *Expert Syst. Appl.*, 36(3):6164–6173.
- Ceder, A. (2016). *Public transit planning and operation: Modeling, practice and behavior*. CRC press.
- Ceder, A. and Wilson, N. H. M. (1986). Bus network design. *Transportation Research Part B: Methodological*, 20(4):331–344.
- Chen, C. (2007). A finite smoothing algorithm for quantile regression. *Journal of Computational and Graphical Statistics*, 16(1):136–164.
- Chen, S. (2019). Quantile regression for duration models with time-varying regressors. *Journal of Econometrics*, 209(1):1–17.
- Cheng, J. (2007). A neural network approach to ordinal regression. *CoRR*, abs/0704.1028.
- Cheng, J.-Y. and Tzeng, S.-J. (2014). Quantile regression of right-censored length-biased data using the buckley-james-type method. *Computational Statistics*, 29(6):1571–1592.
- Chernozhukov, V., Fernández-Val, I., and Kowalski, A. E. (2015). Quantile regression with censoring and endogeneity. *Journal of Econometrics*, 186(1):201–221.
- Chu, K.-F., Lam, A. Y., and Li, V. O. (2019). Deep multi-scale convolutional lstm network for travel demand and origin-destination predictions. *IEEE Transactions on Intelligent Transportation Systems*.
- Chung, Y. and Recker, W. W. (2013). Spatiotemporal analysis of traffic congestion caused by rubbernecking at freeway accidents. *IEEE Trans. Intell. Transp. Syst.*, 14(3):1416–1422.
- Cipriani, E., Gori, S., and Petrelli, M. (2012a). A bus network design procedure with elastic demand for large urban areas. *Public Transport*, 4(1):57–76.
- Cipriani, E., Gori, S., and Petrelli, M. (2012b). Transit network design: A procedure and an application to a large urban area. *Transportation Research Part C: Emerging Technologies*, 20(1):3 – 14.
- Ciuffo, B. and Azevedo, C. L. (2014). A sensitivity-analysis-based approach for the calibration of traffic simulation models. *IEEE Trans. Intell. Transp. Syst.*, 15(3):1298–1309.
- Claeskens, G. and Hjort, N. L. (2008). Model selection and model averaging. *Cambridge Books*.
- Coleman, H. W. and Steele, W. G. (2018). *Experimentation, validation, and uncertainty analysis for engineers*. John Wiley & Sons, 4 edition.
- Corman, F. and Meng, L. (2015). A review of online dynamic models and algorithms for railway traffic management. *IEEE Transactions on Intelligent Transportation Systems*, 16(3):1274–1284.

- Cortés, C. E., Sáez, D., Núñez, A., and Muñoz-Carpintero, D. (2009). Hybrid adaptive predictive control for a dynamic pickup and delivery problem. *Transportation Science*, 43(1):27–42.
- Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., and Bharath, A. A. (2018). Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65.
- D’Agostino, R. B. (1986). *Goodness of Fit Techniques*, volume 68. CRC press.
- Danish Ministry of Transportation (2017). Bred politisk aftale om undersøgelser af nye store vejprojekter. <https://www.trm.dk/da/nyheder/2016/groen-aftale-131216>.
- Danish Road Directorate (2010). Vejregler, trafikteknik, anvendelse af mikrosimuleringsmodeller. http://vejdirektoratet.dk/_layouts/delegate/handlers/GetFile.ashx?id=3944770&filename=Mikrosimuleringsmodeller.pdf.
- Davies, N., Spedding, T., and Watson, W. (1980). Autoregressive moving average processes with non-normal residuals. *Journal of Time Series Analysis*, 1(2):103–109.
- De Jong, G., Daly, A., Pieters, M., Miller, S., Plasmeijer, R., and Hofman, F. (2007). Uncertainty in traffic forecasts: literature review and new results for the netherlands. *Transportation*, 34(4):375–395.
- Dia, H., Gondwe, W., and Panwai, S. (2006). A traffic simulation approach to evaluating the benefits of incident management programs. *29th Australasian Transp. Res. Forum*.
- Digicore Australia (2017). VMS components and functions. <https://digicore-australia.com.au/resources/ivms-components-and-functions/>.
- Donovan, B. and Work, D. (2014). New york city taxi & limousine data. <https://uofi.app.box.com/v/NYCTaxidata/folder/2332218797>. Accessed: 2019-Apr-29.
- DTU Civil Engineering (2019). Weather Data, 02/03/2018–13/03/2019. <http://climatestationdata.byg.dtu.dk>.
- Du, Y., Deng, F., and Liao, F. (2019). A model framework for discovering the spatio-temporal usage patterns of public free-floating bike-sharing system. *Transportation Research Part C: Emerging Technologies*, 103:39 – 55.
- Duda, R. O., Hart, P. E., and Stork, D. G. (2012). *Pattern classification*. John Wiley & Sons.
- Efron, B. (1982). *The jackknife, the bootstrap and other resampling plans*. SIAM.
- EU European Commission (2015). ecall in all new cars from april 2018. <https://ec.europa.eu/digital-single-market/en/news/ecall-all-new-cars-april-2018>.
- Fan, W. and Machemehl, R. B. (2008). A tabu search based heuristic method for the transit route network design problem. In *Computer-aided Systems in Public Transport*, pages 387–408. Springer Berlin Heidelberg.
- Fan Wei and Machemehl Randy B. (2006a). Optimal transit route network design problem with variable transit demand: Genetic algorithm approach. *J. Transp. Eng.*, 132(1):40–51.
- Fan Wei and Machemehl Randy B. (2006b). Using a simulated annealing algorithm to solve the transit route network design problem. *J. Transp. Eng.*, 132(2):122–132.
- Farahani, R. Z., Miandoabchi, E., Szeto, W., and Rashidi, H. (2013). A review of urban transportation network design problems. *European Journal of Operational Research*, 229(2):281 – 302.
- Ferrucci, F., Bock, S., and Gendreau, M. (2013). A pro-active real-time control approach for dynamic vehicle routing problems dealing with the delivery of urgent goods. *European Journal of Operational Research*, 225(1):130–141.
- Finlay, B. and Agresti, A. (1986). *Statistical methods for the social sciences*. Dellen.

- Florida Department of Transportation (2014). Traffic analysis handbook. http://www.fdot.gov/planning/systems/programs/SM/intjus/pdfs/Traffic%20Analysis%20Handbook_March%202014.pdf.
- Freund, D., Henderson, S. G., and Shmoys, D. B. (2019). Bike sharing. In Hu, M., editor, *Sharing Economy: Making Supply Meet Demand*, pages 435–459. Springer International Publishing, Cham.
- Friedman, J. H. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378.
- Galvao, A. F., Lamarche, C., and Lima, L. R. (2013). Estimation of censored quantile regression for panel data with fixed effects. *Journal of the American Statistical Association*, 108(503):1075–1089.
- Gammelli, D., Peled, I., Rodrigues, F., Pacino, D., Kurtaran, H. A., and Pereira, F. C. (2020). Estimating latent demand of shared mobility through censored gaussian processes. *Transportation Research Part C: Emerging Technologies*, 120.
- Gannoun, A., Saracco, J., Yuan, A., and Bonney, G. E. (2005). Non-parametric quantile regression with censored data. *Scandinavian Journal of Statistics*, 32(4):527–550.
- Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian Data Analysis*. Chapman and Hall/CRC, 2nd ed. edition.
- Gelman, A., Jakulin, A., Pittau, M. G., and Su, Y.-S. (2008). A weakly informative default prior distribution for logistic and other regression models. *The annals of applied statistics*, 2(4):1360–1383.
- Genevay, A., Cuturi, M., Peyré, G., and Bach, F. (2016). Stochastic optimization for large-scale optimal transport. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3440–3448.
- Ghahramani, Z. (2015). Probabilistic machine learning and artificial intelligence. Technical report, University of Cambridge.
- Gilks, W. R. (2005). Markov chain monte carlo. In *Encyclopedia of Biostatistics*. American Cancer Society.
- Godfrey, G. A. and Powell, W. B. (2002). An adaptive dynamic programming algorithm for dynamic fleet management, i: Single period travel times. *Transportation Science*, 36(1):21–39.
- Goh, C., Yan, C., and Jaillet, P. (2019). Estimating primary demand in bike-sharing systems. *SSRN Electronic Journal*.
- Gomes, G., May, A., and Horowitz, R. (2004). Calibration of vissim for a congested freeway. Technical report, UC Berkeley.
- Gonzalez, M. C., Hidalgo, C. A., and Barabasi, A.-L. (2008). Understanding individual human mobility patterns. *nature*, 453(7196):779.
- Gordon, A. D., Henzinger, T. A., Nori, A. V., and Rajamani, S. K. (2014). Probabilistic programming. In *Proceedings of the on Future of Software Engineering*, pages 167–181. ACM.
- Gourieroux, C., Monfort, A., Renault, E., and Trognon, A. (1987). Generalised residuals. *Journal of Econometrics*, 34(1-2):5–32.
- Greene, W. (2006). Censored data and truncated distributions. In Mills, T. C. and Patterson, K., editors, *Palgrave Handbook of Econometrics, Volume 1: Econometric Theory*, chapter 20. Palgrave Macmillan, 1 edition.
- Greene, W. H. (2011). Censored data and truncated distributions. *SSRN Electronic Journal*, pages 695–734.
- Grewal, M. S. (2011). Kalman filtering. In Lovric, M., editor, *International Encyclopedia of Statistical Science*, pages 705–708. Springer Berlin Heidelberg, Berlin, Heidelberg.

- Groot, P. and Lucas, P. J. F. (2012). Gaussian process regression with censored data using expectation propagation. In *Proceedings of PGM 2012: the Sixth European Workshop on Probabilistic Graphical Models*.
- Gui, G., Liu, F., Sun, J., Yang, J., Zhou, Z., and Zhao, D. (2019). Flight delay prediction based on aviation big data and machine learning. *IEEE Transactions on Vehicular Technology*, 69(1):140–150.
- Guihaire, V. and Hao, J.-K. (2008). Transit network design and scheduling: A global review. *Transportation Research Part A: Policy and Practice*, 42(10):1251 – 1273.
- Guo, D., Zhu, X., Jin, H., Gao, P., and Andris, C. (2012a). Discovering spatial patterns in origin-destination mobility data. *Transactions in GIS*, 16(3):411–429.
- Guo, F., Krishnan, R., and Polak, J. (2012b). Short-term traffic prediction under normal and incident conditions using singular spectrum analysis and the k-nearest neighbour method. In *IET ITS Conf. Road Transp. Inf. Control*. IET.
- Guo, F., Krishnan, R., and Polak, J. (2014). A novel three-stage framework for short-term travel time prediction under normal and abnormal traffic conditions. In *93rd Transp. Res. Board Annu. Meet.*
- Guo, F., Polak, J. W., and Krishnan, R. (2010). Comparison of modelling approaches for short term traffic prediction under normal and abnormal conditions. In *13th IEEE Conf. Intell. Transp. Syst. (ITSC)*, pages 1209–1214. IEEE.
- Guo, Q. and Karimi, H. A. (2017). A novel methodology for prediction of spatial-temporal activities using latent features. *Computers, Environment and Urban Systems*, 62:74–85.
- Haben, S. and Giasemidis, G. (2016). A hybrid model of kernel density estimation and quantile regression for gecom2014 probabilistic load forecasting. *International Journal of Forecasting*, 32(3):1017–1022.
- Hadjidimitriou, N. S., Lippi, M., and Mamei, M. (2020). A data driven approach to match demand and supply for public transport planning. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–11.
- Hall, P. and Sheather, S. J. (1988). On the distribution of a studentized quantile. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 381–391.
- Harrington, P. (2012). *Machine learning in action*. Manning Publications Co.
- Hashemi, H. and Abdelghany, K. (2015). Real-time traffic network state prediction for proactive traffic management: Simulation experiments and sensitivity analysis. *Transportation Research Record*, 2491(1):22–31.
- Hawas, Y. E. (2007). A microscopic simulation model for incident modeling in urban networks. *Transp. Plan. Technol.*, 30(2-3):289–309.
- He, Y. and Li, H. (2018). Probability density forecasting of wind power using quantile regression neural network and kernel density estimation. *Energy conversion and management*, 164:374–384.
- He, Y., Qin, Y., Wang, S., Wang, X., and Wang, C. (2019). Electricity consumption probability density forecasting method based on lasso-quantile regression neural network. *Applied energy*, 233:565–575.
- He, Y. and Raghunathan, T. E. (2009). On the Performance of Sequential Regression Multiple Imputation Methods with Non Normal Error Distributions. *Communications in Statistics - Simulation and Computation*, 38(4):856–883.
- He, Y. and Zhang, W. (2020). Probability density forecasting of wind power based on multi-core parallel quantile regression neural network. *Knowledge-Based Systems*, 209:106431.
- Henchey, M. J., Batta, R., Blatt, A., Flanigan, M., and Majka, K. (2014). A simulation approach to study emergency response. *J. Simul.*, 8(2):115–128.

- Hensman, J., Fusi, N., and Lawrence, N. D. (2013). Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*.
- Heskes, T. (1997). Practical confidence and prediction intervals. In *Advances in neural information processing systems*, pages 176–182.
- Heywood, P., Maddock, S., Casas, J., Garcia, D., Brackstone, M., and Richmond, P. (2018). Data-parallel agent-based microscopic road network simulation using graphics processing units. *Simul. Model Pract. Theory*, 83:188–200.
- Himanen, V., Nijkamp, P., and Reggiani, A. (2019). *Neural networks in transport applications*. Routledge.
- Ho, S.-L., Xie, M., and Goh, T. N. (2002). A comparative study of neural network and box-jenkins arima modeling in time series prediction. *Computer Ind. Eng.*, 42(2-4):371–375.
- Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.
- Hollander, Y. and Liu, R. (2008). The principles of calibrating traffic microsimulation models. *Transp.*, 35(3):347–362.
- Hoogendoorn, S. P. and Bovy, P. H. (2001). State-of-the-art of vehicular traffic flow modelling. *Proc. Inst. Mech. Eng. I*, 215(4):283–303.
- Horáždovský, P., Novotný, V., and Svítek, M. (2018). Data-driven management of dynamic public transport. In *Smart City Symposium Prague*, pages 1–5. IEEE.
- Hyland, M. and Mahmassani, H. S. (2018). Dynamic autonomous vehicle fleet operations: Optimization-based strategies to assign avs to immediate traveler demand requests. *Transportation Research Part C: Emerging Technologies*, 92:278 – 297.
- Ibarra-Rojas, O., Delgado, F., Giesen, R., and Muñoz, J. (2015). Planning, operation, and control of bus transport systems: A literature review. *Transportation Research Part B: Methodological*, 77:38 – 75.
- Ichoua, S., Gendreau, M., and Potvin, J.-Y. (2006). Exploiting knowledge about future demands for real-time vehicle dispatching. *Transportation Science*, 40(2):211–225.
- Ide, T. and Kato, S. (2009). Travel-time prediction using gaussian process regression: A trajectory-based approach. In *Proceedings of the 2009 SIAM International Conference on Data Mining*, pages 1183–1194.
- Iglesias, R., Rossi, F., Wang, K., Hallac, D., Leskovec, J., and Pavone, M. (2018). Data-driven model predictive control of autonomous mobility-on-demand systems. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–7. IEEE.
- Iliopoulou, C. and Kepaptsoglou, K. (2019). Combining ITS and optimization in public transportation planning: state of the art and future research paths. *European Transport Research Review*, 11(1).
- Ivković, I., Rajić, V., and Stanojević, J. (2020). Coverage probabilities of confidence intervals for the slope parameter of linear regression model when the error term is not normally distributed. *Communications in Statistics - Simulation and Computation*, 49(1):147–158.
- Jackson, D. and White, I. R. (2018). When should meta-analysis avoid making hidden normality assumptions? *Biometrical Journal*, 60(6):1040–1058.
- Jensen, F. V. (1996). *An introduction to Bayesian networks*, volume 210. UCL press London.
- Jia, Y. and Jeong, J.-H. (2020). Deep learning for quantile regression: Deepquantreg.
- Jian, N., Freund, D., Wiberg, H. M., and Henderson, S. G. (2016). Simulation optimization for a large-scale bike-sharing system. In *2016 Winter Simulation Conference (WSC)*, pages 602–613.

- Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233.
- Jordan, M. I. and Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260.
- Karpathy, A. (2015). The unreasonable effectiveness of recurrent neural networks. <http://karpathy.github.io/2015/05/21/rnn-effectiveness/>.
- Katrakazas, C., Quddus, M., and Chen, W. (2018). A simulation study of predicting real-time conflict-prone traffic conditions. *IEEE Trans. Intell. Transp. Syst.*, 19(10):3196–3207.
- Kay, R. (1977). Proportional hazard regression models and the analysis of censored survival data. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 26(3):227–237.
- Kepaptsoglou, K., Asce, M., Karlaftis, M., and Asce, M. (2009). Transit route network design problem: Review. *J. Transp. Eng.*, 135(8):491–505.
- Khan, N., Shahid, S., Juneng, L., Ahmed, K., Ismail, T., and Nawaz, N. (2019). Prediction of heat waves in pakistan using quantile regression forests. *Atmospheric Research*, 221:1–11.
- Khosravi, A., Nahavandi, S., Creighton, D., and Atiya, A. F. (2011). Comprehensive review of neural network-based prediction intervals and new advances. *IEEE Transactions on neural networks*, 22(9):1341–1356.
- Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Koenker, R. (2005). *Quantile Regression*. Cambridge University Press.
- Koenker, R. and Bassett, G. (1978). Regression quantiles. *Econometrica: journal of the Econometric Society*, pages 33–50.
- Koenker, R. and Park, B. J. (1996). An interior point algorithm for nonlinear quantile regression. *Journal of Econometrics*, 71(1-2):265–283.
- Koller, D. and Friedman, N. (2009). *Probabilistic graphical models: principles and techniques*. MIT press.
- Kong, Q.-J., Li, L., Yan, B., Lin, S., Zhu, F., and Xiong, G. (2013). Developing parallel control and management for urban traffic systems. *IEEE Trans. Intell. Transp. Syst.*, 28(3):66–69.
- Koopman, P. and Wagner, M. (2017). Autonomous vehicle safety: An interdisciplinary challenge. *IEEE Intelligent Transportation Systems Magazine*, 9(1):90–96.
- Krishnakumari, P., van Lint, H., Djukic, T., and Cats, O. (2020). A data driven method for od matrix estimation. *Transportation Research Part C: Emerging Technologies*, 113:38–56.
- Kschischang, F. R., Frey, B. J., and Loeliger, H.-A. (2001). Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519.
- Kumar, S. and Toshniwal, D. (2015). A data mining framework to analyze road accident data. *J. Big Data*, 2(1):26.
- Kwon, J., Mauch, M., and Varaiya, P. (2006). Components of congestion: Delay from incidents, special events, lane closures, weather, potential ramp metering gain, and excess demand. *Transp. Res. Rec.*, 1959:84–91.
- Kılıç, F. and Gök, M. (2014). A demand based route generation algorithm for public transit network design. *Computers & Operations Research*, 51:21 – 29.
- Laporte, G., Mesa, J. A., and Perea, F. (2010). A game theoretic framework for the robust railway transit network design problem. *Transportation Research Part B: Methodological*, 44(4):447–459.
- Laporte, G., Meunier, F., and Calvo, R. W. (2018). Shared mobility systems: an updated survey. *Annals of Operations Research*, 271(1):105–126.

- Lee Young-Jae and Vuchic Vukan R. (2005). Transit network design with variable demand. *J. Transp. Eng.*, 131(1):1–10.
- Leng, C. and Tong, X. (2013). A quantile regression estimator for censored data. *Bernoulli*, 19(1):344–361.
- Lewbel, A. and Linton, O. (2002). Nonparametric censored and truncated regression. *Econometrica*, 70(2):765–779.
- Li, A. H. and Bradic, J. (2020). Censored quantile regression forest. In *International Conference on Artificial Intelligence and Statistics*, pages 2109–2119. PMLR.
- Li, J. (2017). Assessing the accuracy of predictive models for numerical data: Not r nor r^2 , why not? then what? *PloS one*, 12(8):e0183250.
- Li, L., Chen, X., and Zhang, L. (2014). Multimodel ensemble for freeway traffic state estimations. *IEEE Trans. Intell. Transp. Syst.*, 15(3):1323–1336.
- Li, R., Pereira, F. C., and Ben-Akiva, M. E. (2018). Overview of traffic incident duration analysis and prediction. *European transport research review*, 10(2):22.
- Li, S., Liu, R., Yang, L., and Gao, Z. (2019). Robust dynamic bus controls considering delay disturbances and passenger demand uncertainty. *Transportation Research Part B: Methodological*, 123:88–109.
- Li, Y., Vinzamuri, B., and Reddy, C. K. (2016). Regularized weighted linear regression for high-dimensional censored data. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 45–53. SIAM.
- Li, Y., Zheng, Y., Zhang, H., and Chen, L. (2015). Traffic prediction in a bike-sharing system. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '15, pages 33:1–33:10. ACM.
- Lin, L., He, Z., and Peeta, S. (2018). Predicting station-level hourly demand in a large-scale bike-sharing network: A graph convolutional neural network approach. *Transportation Research Part C: Emerging Technologies*, 97:258–276.
- LINC (2020). The largest test of self-driving shuttles in denmark. <http://lincproject.dk/en/om/>.
- Liu, S., Yue, Y., and Krishnan, R. (2013). Adaptive Collective Routing Using Gaussian Process Dynamic Congestion Models. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 704–712, New York, NY, USA. ACM.
- Lou, Y., Yin, Y., and Lawphongpanich, S. (2009). Robust approach to discrete network designs with demand uncertainty. *Transp. Res. Rec.*, 2090(1):86–94.
- Lu, Y., Seshadri, R., Pereira, F., OSullivan, A., Antoniou, C., and Ben-Akiva, M. (2015). Dymit2.0: Architecture design and preliminary results on real-time data fusion for traffic prediction and crisis management. In *18th Int Conf Intell Transp Syst*, pages 2250–2255. IEEE.
- Lund Andersen, C. (2016). Evaluation summary of the pilot trial with hard shoulder running on the hillerød motorway. Technical report, Danish Road Directorate.
- Mak, T. K. (2000). Heteroscedastic regression models with non-normally distributed errors. *Journal of Statistical Computation and Simulation*, 67(1):21–36.
- Manjunatha, P., Vortisch, P., and Mathew, T. V. (2013). Methodology for the calibration of vissim in mixed traffic. In *92nd Transp. Res. Board Annu. Meet.*
- Maze, T. H., Agarwal, M., and Burchett, G. (2006). Whether weather matters to traffic demand, traffic safety, and traffic operations and flow. *Transportation research record*, 1948(1):170–176.
- Mazloumi, E., Rose, G., Currie, G., and Moridpour, S. (2011). Prediction intervals to account for uncertainties in neural network predictions: Methodology and application in bus travel time prediction. *Engineering Applications of Artificial Intelligence*, 24(3):534–542.

- Meneses, F. and Moreira, A. (2012). Large scale movement analysis from wifi based location data. In *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, pages 1–9. IEEE.
- Miller, J. and How, J. P. (2017). Predictive positioning and quality of service ridesharing for campus mobility on demand systems. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1402–1408. IEEE.
- Minka, T. P. (2001). Expectation propagation for approximate bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 362–369. Morgan Kaufmann Publishers Inc.
- Mir, Z. H. and Filali, F. (2016). An adaptive kalman filter based traffic prediction algorithm for urban road network. In *12th Int Conf Innov Inf Tech*, pages 1–6.
- Mroz, T. A. (1987). The sensitivity of an empirical model of married women’s hours of work to economic and statistical assumptions. *Econometrica: Journal of the econometric society*, pages 765–799.
- Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.
- Murphy, R. R. (2019). *Introduction to AI robotics*. MIT press.
- Nagy, A. M. and Simon, V. (2018). Survey on traffic prediction in smart cities. *Pervasive and Mobile Computing*, 50:148–163.
- Narayanan, S., Chaniotakis, E., and Antoniou, C. (2020). Shared autonomous vehicle services: A comprehensive review. *Transportation Research Part C: Emerging Technologies*, 111:255 – 293.
- Nasrabadi, N. M. (2007). Pattern recognition and machine learning. *Journal of electronic imaging*, 16(4):049901.
- Nelson, H. and Granger, C. (1979). Experience with using the box-cox transformation when forecasting economic time series. *Journal of Econometrics*, 10(1):57–69.
- Ng, A. Y. (2004). Feature selection, l_1 vs. l_2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78.
- Ni, M., He, Q., and Gao, J. (2014). Using social media to predict traffic flow under special event conditions. In *93rd Transp. Res. Board Annu. Meet.*
- Nielsen, M. A. (2015). *Neural networks and deep learning*, volume 2018. Determination press San Francisco, CA.
- Nilsson, N. J. (1965). *Learning Machines: Foundations of Trainable Pattern-Classifying Systems*. The McGraw-Hill Companies.
- Nosratabadi, S., Mosavi, A., Keivani, R., Ardabili, S., and Aram, F. (2019). State of the art survey of deep learning and machine learning models for smart cities and urban sustainability. In *International Conference on Global Research and Education*, pages 228–238. Springer.
- NYC (2016). Nyc open data. <https://opendata.cityofnewyork.us/>.
- Olson, R. S., La Cava, W., Mustahsan, Z., Varik, A., and Moore, J. H. (2017). Data-driven advice for applying machine learning to bioinformatics problems. *arXiv preprint arXiv:1708.05070*.
- O’Mahony, E. and Shmoys, D. B. (2015). Data analysis and optimization for (citi)bike sharing. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, AAAI’15*, pages 687–694. AAAI Press.
- Owens, N., Armstrong, A., Sullivan, P., Mitchell, C., Newton, D., Brewster, R., and Trego, T. (2010). Traffic incident management handbook. Technical report, Sci. Appl. Int. Corp.
- Pan, S. J. and Yang, Q. (2009). A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359.

- Park, B. and Won, J. (2006). Microscopic simulation model calibration and validation handbook. Technical report, Virginia Transp. Res. Council. 07-CR6.
- Pearl, J. (2014). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier.
- Peled, I., Kamalakara, R., Lima Azevedo, C., and Pereira, F. (2020). Qtip: Quick simulation-based adaptation of traffic model per incident parameters. *Journal of Simulation*.
- Peled, I., Lee, K., Jiang, Y., Dauwels, J., and Pereira, F. (2019a). Preserving uncertainty in demand prediction for autonomous mobility services. In *Proceedings of the IEEE Intelligent Transportation Systems Conference (ITSC) 2019*, pages 3043–3048, United States. IEEE.
- Peled, I., Lee, K., Jiang, Y., Dauwels, J., and Pereira, F. C. (2019b). Online predictive optimization framework for stochastic demand-responsive transit services. *arXiv preprint arXiv:1902.09745*.
- Peled, I., Pereira, F. C., and Winther, O. (2018). Traffic prediction with convolutional long short-term memory. In *7th Symposium of the European Association for Research in Transportation (hEART2018)*.
- Peled, I., Rodrigues, F., and Pereira, F. C. (2019c). Model-based machine learning for transportation. In *Mobility Patterns, Big Data and Transport Analytics*, pages 145–171. Elsevier.
- Pereira, F. C., Antoniou, C., Fargas, J. A., and Ben-Akiva, M. (2014). A metamodel for estimating error bounds in real-time traffic prediction systems. *IEEE Transactions on Intelligent Transportation Systems*, 15(3):1310–1322.
- Pereira, F. C. and Rodrigues, F. (2018). Model-based machine learning. <https://http://kurser.dtu.dk/course/42186/>.
- Pereira, F. C., Rodrigues, F., and Ben-Akiva, M. (2013). Text analysis in incident duration prediction. *Transportation Research Part C: Emerging Technologies*, 37:177–192.
- Pernot, P., Huang, B., and Savin, A. (2020). Impact of non-normal error distributions on the benchmarking and ranking of quantum machine learning models. *Machine Learning: Science and Technology*.
- Petersen, N. C., Rodrigues, F., and Pereira, F. C. (2019). Multi-output bus travel time prediction with convolutional lstm neural network. *Expert Systems with Applications*, 120:426–435.
- Powell, J. L. (1984). Least absolute deviations estimation for the censored regression model. *Journal of Econometrics*, 25(3):303–325.
- Powell, J. L. (1986). Censored regression quantiles. *Journal of econometrics*, 32(1):143–155.
- Powell, W. B. (1988). A comparative review of alternative algorithms for the dynamic vehicle allocation problem. In L., G. B. and A., A. A., editors, *Vehicle Routing: Methods and Studies*, pages 249–291. Elsevier, Amsterdam, The Netherlands.
- Powell, W. B. (1996). A stochastic formulation of the dynamic assignment problem, with an application to truckload motor carriers. *Transportation Science*, 30(3):195–219.
- Powell, W. B. and Topaloglu, H. (2003). Stochastic programming in transportation and logistics. *Handbooks in operations research and management science*, 10:555–635.
- Prakash, A. A., Seshadri, R., Antoniou, C., Pereira, F. C., and Ben-Akiva, M. (2018). Improving scalability of generic online calibration for real-time dynamic traffic assignment systems. *Transp. Res. Rec.*, 2672:79–92.
- Profillidis, V. and Botzoris, G. (2019). Chapter 3 - methods of modeling transport demand. In Profillidis, V. and Botzoris, G., editors, *Modeling of Transport Demand*, pages 89 – 123. Elsevier.
- PTV Group (2017). Vissim 8 user manual. http://cgi.ptvgroup.com/cgi-bin/en/traffic/vissim_download.pl.

- Qin, X. and Mahmassani, H. (2004). Adaptive calibration of dynamic speed-density relations for online network traffic estimation and prediction applications. *Transp. Res. Rec.*, 1876:82–89.
- Quiñonero-Candela, J., Rasmussen, C. E., Sinz, F., Bousquet, O., and Schölkopf, B. (2006). Evaluating predictive uncertainty challenge. In Quiñonero-Candela, J., Dagan, I., Magnini, B., and d’Alché Buc, F., editors, *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Textual Entailment*, pages 1–27, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.
- Rajkomar, A., Dean, J., and Kohane, I. (2019). Machine learning in medicine. *New England Journal of Medicine*, 380(14):1347–1358.
- Rasmussen, C. E. (2003). Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer.
- Rasmussen, C. E. and Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.
- Rasouli, S. and Timmermans, H. (2012). Uncertainty in travel demand forecasting models: literature review and research agenda. *Transportation letters*, 4(1):55–73.
- Rejsekort (2020). About rejsekort & rejseplan a/s. <https://www.rejsekort.dk/en/rkrp>.
- Rich, J. and Vandet, C. A. (2019). Is the value of travel time savings increasing? analysis throughout a financial crisis. *Transportation Research Part A: Policy and Practice*, 124:145–168.
- Rixey, R. A. (2013). Station-level forecasting of bikesharing ridership: Station network effects in three u.s. systems. *Transportation Research Record*, 2387(1):46–55.
- Roberts, S., Osborne, M., Ebden, M., Reece, S., Gibson, N., and Aigrain, S. (2013). Gaussian processes for time-series modelling. *Philos. Trans. R. Soc. A*, 371(1984):20110550.
- Rodrigues, F., Borysov, S. S., Ribeiro, B., and Pereira, F. C. (2017). A Bayesian Additive Model for Understanding Public Transport Usage in Special Events. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(11):2113–2126.
- Rodrigues, F., Henrickson, K., and Pereira, F. C. (2018). Multi-output gaussian processes for crowdsourced traffic data imputation. *IEEE Transactions on Intelligent Transportation Systems*, 20(2):594–603.
- Rodrigues, F., Markou, I., and Pereira, F. C. (2019). Combining time-series and textual data for taxi demand prediction in event areas: A deep learning approach. *Information Fusion*, 49:120–129.
- Rodrigues, F. and Pereira, F. C. (2018). Heteroscedastic gaussian processes for uncertainty modeling in large-scale crowdsourced traffic data. *Transportation research part C: emerging technologies*, 95:636–651.
- Rodrigues, F. and Pereira, F. C. (2020). Beyond expectation: deep joint mean and quantile regression for spatiotemporal problems. *IEEE Transactions on Neural Networks and Learning Systems*.
- Rodrigues, F. M. P. D. (2016). *Probabilistic models for learning from crowdsourced data*. PhD thesis, University of Coimbra.
- Rudloff, C. and Lackner, B. (2014). Modeling demand for bikesharing systems: Neighboring stations as source for demand and reason for structural breaks. *Transportation Research Record*, 2430(1):1–11.
- Salamanis, A., Margaritis, G., Kehagias, D. D., Matzoulas, G., and Tzovaras, D. (2017). Identifying patterns under both normal and abnormal traffic conditions for short-term traffic prediction. *Transp. Res. Procedia*, 22:665–674.

- Saltelli, A., Tarantola, S., Campolongo, F., and Ratto, M. (2004). *Sensitivity analysis in practice: a guide to assessing scientific models*. Wiley Online Library, 1 edition.
- Sangnier, M., Fercoq, O., and d'Alché Buc, F. (2016). Joint quantile regression in vector-valued rkhs. In *Advances in Neural Information Processing Systems*, pages 3693–3701.
- Santos, G. (2018). Sustainability and shared mobility models. *Sustainability*, 10(9):3194.
- Schlögl, M. and Stütz, R. (2019). Methodological considerations with data uncertainty in road safety analysis. *Accident Analysis & Prevention*, 130:136 – 150.
- Schuster-Böckler, B. and Bateman, A. (2007). An introduction to hidden markov models. *Current protocols in bioinformatics*, 18(1):A–3A.
- Seber, G. A. and Lee, A. J. (2012). *Linear regression analysis*, volume 329. John Wiley & Sons.
- Seo, Y., Defferrard, M., Vandergheynst, P., and Bresson, X. (2016). Structured sequence modeling with graph convolutional recurrent networks. *CoRR*, abs/1612.07659.
- Sevtsuk, A. (2009). Mapping the mit campus in real time using wifi. In *Handbook of Research on Urban Informatics: The Practice and Promise of the Real-Time City*, pages 326–338. IGI Global.
- Shao, H., Lam, W. H., Sumalee, A., Chen, A., and Hazelton, M. L. (2014). Estimation of mean and covariance of peak hour origin–destination demands from day-to-day traffic counts. *Transportation Research Part B: Methodological*, 68:52–75.
- Shi, X., Chen, Z., Wang, H., Yeung, D., Wong, W., and Woo, W. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *CoRR*, abs/1506.04214.
- Shim, J. and Hwang, C. (2009). Support vector censored quantile regression under random censoring. *Computational statistics & data analysis*, 53(4):912–919.
- Siegel, J. E., Erb, D. C., and Sarma, S. E. (2018). A survey of the connected vehicle landscape: Architectures, enabling technologies, applications, and development areas. *IEEE Trans. Intell. Transp. Syst.*, 19(8):2391–2406.
- Silverman, B. W. (1985). Some aspects of the spline smoothing approach to non-parametric regression curve fitting. *Journal of the Royal Statistical Society. Series B (Methodological)*, 47(1):1–52.
- Singhvi, D., Singhvi, S., Frazier, P. I., Henderson, S. G., O’Mahony, E., Shmoys, D. B., and Woodard, D. B. (2015). Predicting bike usage for new york city’s bike sharing system. In *AAAI Workshop: Computational Sustainability*.
- Smith, B. L., Williams, B. M., and Oswald, R. K. (2002). Comparison of parametric and nonparametric models for traffic flow forecasting. *Transp. Res. Part C Emerg. Technol.*, 10(4):303–321.
- Smith, G. and Hensher, D. A. (2020). Towards a framework for mobility-as-a-service policies. *Transport policy*, 89:54–65.
- Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. In *Adv. Neural Inf. Process. Syst.*, pages 2951–2959.
- Solomatine, D. P. and Shrestha, D. L. (2009). A novel method to estimate model uncertainty using machine learning techniques. *Water Resources Research*, 45(12).
- Spivey, M. Z. and Powell, W. B. (2004). The dynamic assignment problem. *Transportation Science*, 38(4):399–419.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.
- Stein, D. M. (1978). Scheduling dial-a-ride transportation systems. *Transportation Science*, 12(3):232–249.

- Szeto, W. and Jiang, Y. (2014). Transit route and frequency design: Bi-level modeling and hybrid artificial bee colony algorithm approach. *Transportation Research Part B: Methodological*, 67:235 – 263.
- Szeto, W. Y. and Wu, Y. (2011). A simultaneous bus route design and frequency setting problem for tin shui wai, hong kong. *Eur. J. Oper. Res.*, 209(2):141–155.
- Taylor, J. W. (2000). A quantile regression neural network approach to estimating the conditional density of multiperiod returns. *Journal of Forecasting*, 19(4):299–311.
- Team, S. D. (2018). Stan modeling language users guide and reference manual. <https://mc-stan.org>.
- Terza, J. V. (1985). A tobit-type estimator for the censored poisson regression model. *Economics Letters*, 18(4):361–365.
- Thode, H. C. (2002). *Testing for normality*, volume 164. CRC press.
- Tian, W., Song, J., Guo, Y., and Yang, F. (2020). Probabilistic airport traffic demand prediction incorporating the weather factors. In *Journal of Physics: Conference Series*, volume 1510, page 012022.
- Tipping, M. E. and Bishop, C. M. (1999). Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622.
- Titsias, M. (2009). Variational learning of inducing variables in sparse gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574.
- Tobin, J. (1958). Estimation of Relationships for Limited Dependent Variables. *Econometrica*, 26(1):24–36.
- Toman, P., Zhang, J., Ravishanker, N., and Konduri, K. C. (2020). Dynamic predictive models for ridesourcing services in new york city using daily compositional data. *Transportation Research Part C: Emerging Technologies*, 121:102833.
- Toole, J. L., Colak, S., Sturt, B., Alexander, L. P., Evsukoff, A., and González, M. C. (2015). The path most traveled: Travel demand estimation using big data resources. *Transportation Research Part C: Emerging Technologies*, 58:162 – 177.
- Tsai, T.-H., Lee, C.-K., and Wei, C.-H. (2009). Neural network based temporal feature models for short-term railway passenger demand forecasting. *Expert Systems with Applications*, 36(2):3728–3736.
- Tune, P., Roughan, M., and Cho, K. (2016). A comparison of information criteria for traffic model selection. In *10th Intern Conf Signal Process Commun Sys (ICSPCS)*, pages 1–10. IEEE.
- Tupper, L. L., Chowdhury, M. A., Klotz, L., and Fries, R. N. (2012). Measuring sustainability: How traffic incident management through intelligent transportation systems has greater energy and environmental benefits than common construction-phase strategies for "green" roadways. *Int. J. Sustain. Transp.*, 6(5):282–297.
- U.S. Department of Commerce (2018). National oceanic and atmospheric administration. <http://www.noaa.gov/>.
- van Hentenryck, P. and Bent, R. (2009). *Online stochastic combinatorial optimization*. The MIT Press.
- van Hinsbergen, C. P. I. J., van Lint, J. W. C., and van Zuylen, H. (2009). Bayesian committee of neural networks to predict travel times with confidence intervals. *Transportation Research Part C: Emerging Technologies*, 17(5):498–509.
- van Lint, J. W. C., Hoogendoorn, S. P., and van Zuylen, H. J. (2005). Accurate freeway travel time prediction with state-space neural networks under missing data. *Transportation Research Part C: Emerging Technologies*, 13(5-6):347–369.
- van Lint, J. W. C., van Zuylen, H. J., and Tu, H. (2008). Travel time unreliability on freeways: Why measures based on variance tell only half the story. *Transportation Research Part A: Policy and Practice*, 42(1):258–277.

- Viereckl, R., Ahlemann, D., and Koster, A. (2016). Connected car report 2016. <https://www.strategyand.pwc.com/gx/en/insights/2016/connected-car-2016-study.html>.
- Vlahogianni, E., Karlaftis, M., Golias, J., and Halkias, B. (2010). Freeway operations, spatiotemporal-incident characteristics, and secondary-crash occurrence. *Transp. Res. Rec.*, 2178:1–9.
- Wainwright, M. J. and Jordan, M. I. (2008). *Graphical models, exponential families, and variational inference*. Now Publishers Inc.
- Wan, E. (2006). Sigma-point filters: an overview with applications to integrated navigation and vision assisted control. In *2006 IEEE Nonlinear Statistical Signal Processing Workshop*, pages 201–202. IEEE.
- Wang, F.-Y. (2010). Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications. *IEEE Trans. Intell. Transp. Syst.*, 11(3):630–638.
- Wang, H., Lam, W. H. K., Zhang, X., and Shao, H. (2015). Sustainable transportation network design with stochastic demands and chance constraints. *International Journal of Sustainable Transportation*, 9(2):126–144.
- Wang, H. and Yeung, D.-Y. (2016). Towards bayesian deep learning: A framework and some existing methods. *IEEE Transactions on Knowledge and Data Engineering*, 28(12):3395–3408.
- Wang, K., Pleiss, G., Gardner, J., Tyree, S., Weinberger, K. Q., and Wilson, A. G. (2019). Exact gaussian processes on a million data points. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 14648–14659. Curran Associates, Inc.
- Wang, W., Zhang, H., Li, T., Guo, J., Huang, W., Wei, Y., and Cao, J. (2020). An interpretable model for short term traffic flow prediction. *Math. Comput. Simul.*, 171:264–278.
- Wang, X. and Qing-dao-er ji, R. (2019). Application of optimized genetic algorithm based on big data in bus dynamic scheduling. *Cluster Computing*, 22(6):15439–15446.
- Wei, L.-J. (1992). The accelerated failure time model: a useful alternative to the cox regression model in survival analysis. *Statistics in medicine*, 11(14-15):1871–1879.
- Wei, S. X. (1999). A bayesian approach to dynamic tobit models. *Econometric Reviews*, 18(4):417–439.
- Weisberg, S. (2005). *Applied linear regression*, volume 528. John Wiley & Sons.
- Wen, J., Zhao, J., and Jaillet, P. (2017). Rebalancing shared mobility-on-demand systems: A reinforcement learning approach. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 220–225. IEEE.
- Williams, C. K. and Rasmussen, C. E. (2006). *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA.
- Wirtz, J., Schofer, J., and Schulz, D. (2005). Using simulation to test traffic incident management strategies: The benefits of preplanning. *Transp. Res. Rec.*, 1923:82–90.
- Wold, S., Esbensen, K., and Geladi, P. (1987). Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52.
- Wolters, R. and Kateman, G. (1989). The performance of least squares and robust regression in the calibration of analytical methods under non-normal noise distributions. *Journal of chemometrics*, 3(2):329–342.
- Woelf, P., Burge, C., Keating, A., and Yaffe, M. (2004). Statistics and probability primer for computational biologists. *Massachusetts Institute of Technology*.
- Wu, T., Xie, K., Xinpin, D., and Song, G. (2012). An online boosting approach for traffic flow forecasting under abnormal conditions. In *9th Int. Conf. Fuzzy Syst. Knowl. Discov.*, pages 2555–2559. IEEE.

- Wu, W., Yeh, M.-Y., and Chen, M.-S. (2018). Deep censored learning of the winning price in the real time bidding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2526–2535. ACM.
- Xie, Y., Zhao, K., Sun, Y., and Chen, D. (2010). Gaussian Processes for Short-Term Traffic Volume Forecasting. *Transportation Research Record*, 2165(1):69–78.
- Xu, C., Ji, J., and Liu, P. (2018). The station-free sharing bike demand forecasting with a deep learning approach and large-scale datasets. *Transportation Research Part C: Emerging Technologies*, 95:47 – 60.
- Xue, R., Sun, D. J., and Chen, S. (2015). Short-term bus passenger demand prediction based on time series model and interactive multiple model approach. *Discrete Dynamics in Nature and Society*, 2015.
- Yang, C., Chen, A., Xu, X., and Wong, S. (2013). Sensitivity-based uncertainty analysis of a combined travel demand model. *Transportation Research Part B: Methodological*, 57:225–244.
- Yang, Y., Fan, Y., and Royset, J. O. (2019). Estimating probability distributions of travel demand on a congested network. *Transportation Research Part B: Methodological*, 122:265–286.
- Yang, Y., Li, S., Li, W., and Qu, M. (2018). Power load probability density forecasting using gaussian process quantile regression. *Applied Energy*, 213:499–509.
- Yang, Y., Wang, H. J., and He, X. (2016a). Posterior inference in bayesian quantile regression with asymmetric laplace likelihood. *International Statistical Review*, 84(3):327–344.
- Yang, Z., Hu, J., Shu, Y., Cheng, P., Chen, J., and Moscibroda, T. (2016b). Mobility modeling and prediction in bike-sharing systems. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '16, pages 165–178. ACM.
- Yu, K. and Stander, J. (2007). Bayesian analysis of a Tobit quantile regression model. *Journal of Econometrics*, 137(1):260–276.
- Zhang, C., Osorio, C., and Flötteröd, G. (2017). Efficient calibration techniques for large-scale traffic simulators. *Transp. Res. Part B Meth.*, 97:214–239.
- Zhang, J., Wang, F., Wang, K., Lin, W., Xu, X., and Chen, C. (2011). Data-driven intelligent transportation systems: A survey. *IEEE Trans. Intell. Transp. Syst.*, 12(4):1624–1639.
- Zhang, J., Zheng, Y., and Qi, D. (2016). Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Proceeding of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*.
- Zhang, W., Quan, H., and Srinivasan, D. (2018). An improved quantile regression neural network for probabilistic load forecasting. *IEEE Transactions on Smart Grid*, 10(4):4425–4434.
- Zhang, Y. and Haghani, A. (2015). A gradient boosting method to improve travel time prediction. *Transp. Res. Part C Emerg. Technol.*, 58:308–324.
- Zhang, Y. and Yang, Q. (2018). An overview of multi-task learning. *National Science Review*, 5(1):30–43.
- Zheng, S. (2012). Qboost: Predicting quantiles with boosting for regression and binary classification. *Expert Systems with Applications*, 39(2):1687–1697.
- Zheng, X., Chen, W., Wang, P., Shen, D., Chen, S., Wang, X., Zhang, Q., and Yang, L. (2016). Big data for social transportation. *IEEE Trans. Intell. Transp. Syst.*, 17(3):620–630.

Model-Based Machine Learning for Transportation

In my first year of PhD studies, I wrote this Chapter (Peled et al., 2019c) as part of the book “Mobility Patterns, Big Data and Transport Analytics: Tools and Applications for Modeling” (Antoniou et al., 2018). The Chapter is mostly a summary of material from DTU course “Model-Based Machine Learning” by the co-authors: Assoc. Prof. Filipe Rodrigues and my main supervisor, Prof. Francisco C. Pereira (Pereira and Rodrigues, 2018), with a few parts adapted from the PhD thesis of Rodrigues (2016).

In the Model-Based approach to Machine Learning (MBML), the first step is to formulate the uncertainty structure of the problem as a probabilistic model. The next step is to perform Bayesian Inference, which updates the distribution of unknown parameters in the model per observed evidence. This Chapter rigorously defines these steps and demonstrates the advantages of MBML through several case studies in the Transportation domain.

Writing a book chapter was a rewarding experience, through which I became better familiar with Bayesian Inference and probabilistic modeling, as well as their advantages and shortcomings vs. the classic, Frequentist approach to modeling. I have since applied these useful concepts in several other works, e.g., as in Chapters 2 and 3.

A.1 Introduction

The world around us is rich with *uncertainty*, possibly because of innate non-determinism (Koller and Friedman, 2009). In the transportation domain alone, one finds uncertainty in e.g. road safety, train delays, bus dwell times, or passengers’ choice of travel mode. Models of reality must therefore account for such prevalent uncertainty. Consequently, at the heart of modern Machine Learning is *probability theory*, which provides a consistent framework for quantifying and manipulating uncertainty.

In the Model-Based approach to Machine Learning (MBML), uncertainty is represented and manipulated through the coherent use of probability theory. When given data for modeling, the first step in the MBML approach is to define our assumptions about the stochastic process which generated the data. MBML is thus antithetical to traditional Machine Learning methods, where modeling

Concepts	Relevance	Reference
Probability Distribution: continuous, discrete, univariate, multivariate; Probability Density Function (PDF); Random Variables, (conditional) independence.	Prerequisite	(Bishop, 2006) Section 1.2.1
Joint, marginal and conditional distributions, sum and product rules of probability; Likelihood function, prior and posterior distributions, Bayes' rule.	Recommended	(Bishop, 2006) Section 1.2
Gaussian (Normal), Poisson, categorical, Dirichlet and Cauchy distributions.	Recommended	(Woolf et al., 2004)

Table A.1: Probability theory concepts used in this chapter.

begins with the dilemma of picking an algorithm to learn patterns in the data (Olson et al., 2017). Furthermore, MBML provides a mathematically grounded framework for measuring how uncertainty changes when new observations become available.

We introduce MBML through four case studies, which demonstrate how MBML elegantly treats a wide range of common modeling problems. The first case study deals with a *continuous regression problem*, and introduces the key components of MBML. It also defines *Bayesian inference*, through which we measure the certainty gain from observed evidence. The second case study deals with a *discrete classification problem*. The third case study handles a *time series problem* through the large class of State-Space Models. The fourth and last case study introduces *topic modeling* for numerically encoding *textual data* in regression problems. Each case study starts by describing an actual dataset, then detaches from specific data details, and proceeds to build useful models for the general problem domain.

We assume the reader is familiar with fundamentals of linear algebra and basic concepts and notation of probability theory, as instructed in Table A.1. Further, this chapter uses the following mathematical notation. Letters which are not bold (e.g. y , D or σ) denote scalars. A lower case bold letter (e.g. $\mathbf{x}$ or β) denotes a vector, while an upper case bold letter (e.g. $\mathbf{X}$ or Σ) denotes a matrix. $\mathbf{0}$ and $\mathbf{I}$ denote, respectively, a zero square matrix and an identity matrix, with dimensions determined by context.

A.2 Case Study 1: Taxi Demand in New York City

As a first case study, we model taxi demand in New York City. Such a model is useful for e.g. optimizing taxi services, so that taxis can be proactively dispatched to locations where high demand is anticipated. More generally, this model is applicable to various problems of demand modeling, such as public or shared transport, or allocation of resources, such as water, energy, or communication. We use this case study to also introduce some of the key components of MBML.

Our data consists of two datasets: taxi pickups around Wall Street in years 2009 to 2015 (NYC, 2016), and corresponding weather information from the National Oceanic and Atmospheric Administration (U.S. Department of Commerce, 2018), such as wind speed, pressure, and precipitation. The data is given as a matrix, where each row consists of hourly aggregated pickups and weather information. Each of the $N = 8760$ matrix rows consists of $L = 16$ *data features* (date, hour of day, and weather), followed by the corresponding value of the *target variable* (number of taxi pickups).

Our goal is to model the dependency of the target variable on the data features. We thus partition the data matrix as a sub-matrix $\mathbf{X}^{N \times L}$ of data features, and a column vector $\mathbf{y}^{N \times 1} = (y_1, \dots, y_N)^T$ of corresponding pickup counts. We next assume that for all $n = 1, \dots, N$, y_n depends on $\mathbf{x}_n$, the n 'th row of $\mathbf{X}$, as

$$y_n = f(\mathbf{x}_n) + \varepsilon, \quad (\text{A.1})$$

where f is an unknown function, and ε is error (“noise”) in the observations. This case study thus demonstrates a *regression problem*, where we wish to find a function f that models the dependency in (A.1) well.

A.2.1 Initial Probabilistic Model: Linear Regression

Linear Regression is a common starting point in regression problems, and serves as a simple foundation for building powerful models. Our initial model M_1 is a linear regression model, whereby we assume there exists a vector of coefficients $\beta = (\beta_1, \dots, \beta_L)^T \in \mathbb{R}^L$, so that

$$y_n = \beta^T \mathbf{x}_n + \varepsilon. \quad (\text{A.2})$$

Note that we chose not to include a *zero-intercept* (“*bias*”) coefficient β_0 in our model. If desired, we can nevertheless include β_0 along with a corresponding data feature, which is fixed as 1.

A.2.1.1 Likelihood Function

Next, we assume that ε is *white noise*, i.e. ε is normally distributed as $\varepsilon \sim \mathcal{N}(\varepsilon | 0, \sigma^2)$ for some unknown $\sigma \in \mathbb{R}$. This too is a common initial assumption. Equivalently, we have modeled $\mathbf{y}$ as a random variable, so that

$$y_n \sim \mathcal{N}(y_n | \beta^T \mathbf{x}_n, \sigma^2). \quad (\text{A.3})$$

As such, we assume that all observations are independent samples from a multivariate Gaussian, which has mean $\beta^T \mathbf{X}$ and variance σ^2 . For any given value of β and σ , the mean and variance are thus known and fixed, and the *likelihood* of obtaining the particular observations $\mathbf{y}$ is

$$p(\mathbf{y} | \mathbf{X}, \beta, \sigma) = \prod_{n=1}^N \mathcal{N}(y_n | \beta^T \mathbf{x}_n, \sigma^2), \quad (\text{A.4})$$

where

$$\mathcal{N}(y_n | \beta^T \mathbf{x}_n, \sigma^2) = \frac{\exp\left(-\left(y_n - \beta^T \mathbf{x}_n\right)^2 / 2\sigma^2\right)}{\sqrt{2\pi\sigma^2}} \quad (\text{A.5})$$

is the Probability Density Function (PDF) of the Normal distribution.

A.2.1.2 Priors

Let us now assume for simplicity that σ is known in advance in our model, so that we are only interested in learning parameter β , the linear coefficients. MBML invites us to encode any prior knowledge we have about the model parameters as a *prior* distribution. We thus assign β a prior distribution, as

$$\beta \sim \mathcal{N}(\beta | \mathbf{0}, \lambda \mathbf{I}). \quad (\text{A.6})$$

That is, our prior belief is that each linear coefficient is distributed identically and independently (“i.i.d”) as a Gaussian with mean zero and variance λ , which is now an additional component of our model. For simplicity, we fix λ , carefully choosing its value: too large variance renders the prior distribution ineffective, which leads to overfitting, whereas too low variance overly restricts the coefficients, which leads to underfitting. This is an instance of *regularization* for prevention of overfitting and underfitting (Ng, 2004).

A.2.2 Key Components of MBML

Our initial, probabilistic linear regression model M_1 is now complete. As we shall soon explain, our goal now is to obtain the *posterior distribution* of model

Given $\mathbf{X}, \sigma, \lambda$:

1. Draw $\beta \sim \mathcal{N}(\beta \mid \mathbf{0}, \lambda \mathbf{I})$.
2. For each feature vector $\mathbf{x}_n$, $n = 1 \dots N$:
 - 2.1. Draw target $y_n \sim \mathcal{N}(y_n \mid \beta^T \mathbf{x}_n, \sigma^2)$.

Figure A.1: Generative process for linear regression model M_1

parameters, namely their distribution after (“a-posteriori”) observing the evidence $\mathbf{y}$. MBML facilitates this goal through three key components: 1) generative process, 2) Probabilistic Graphical Model, and 3) factorized joint probability distribution, all of which we describe next.

A.2.2.1 Generative Process

We have established our assumptions on the structure and operation of M_1 by reasoning about a possible chain of random events, which yielded the observations $\mathbf{y}$ from the data features $\mathbf{X}$. That is, we imagined a *generative process*, through which the given observations manifested. Establishing a generative process clarifies our initial assumptions about the structure of uncertainty in the problem, and is therefore a first step in designing any MBML model. Figure A.1 summarizes the generative process of M_1 in concise, algorithmic form.

A.2.2.2 Probabilistic Graphical Model (PGM)

Probabilistic Graphical Models (PGMs) describe the structure of probabilistic models intuitively and compactly. A PGM is a graph, where nodes represent random variables, and edges represent probabilistic relationship. In this chapter, we will use only directed PGMs, also known as Bayesian networks, where edge directions represent causality (Jensen, 1996).

Let us describe the *components of PGMs* by reviewing Figure A.2a on the next page, which provides a PGM for M_1 . Large circles are nodes, annotated with the corresponding variables. Nodes for observed variables are shaded, whereas nodes for unobserved (“latent”) variables are unshaded. The plate marked with N stands for N repeated applications, i.e. it describes the relationship between $\mathbf{x}_n$ and y_n for all $n = 1..N$. Small black circles represent pre-fixed (hyper-)parameters, which are sometimes not shown, to reduce visual clutter. Note that a PGM does not specify the underlying probability distributions, hence

Figure A.2: Probabilistic Graphical Models for case study 1: (a) homoscedastic linear regression, (b) heteroscedastic linear regression, (c) Poisson regression. Hyper-parameters are not shown in (b) and (c).

the same PGM may apply to many different probabilistic models.

A.2.2.3 Joint Probability Distribution

Through the sum rule of probability (Bishop, 2006), the *joint probability distribution* of all variables in a probabilistic model can yield the marginal distribution of any subset of these variables. Therefore, the joint probability distribution is the central object for calculating the updated state of uncertainty, given evidence. It is thus beneficial to *factorize* the joint probability distribution as much as possible. For M_1 , the joint probability distribution of β and $\mathbf{y}$, given all known variables, factorizes as following.

$$\begin{aligned}
 p(\beta \mathbf{y} \mid \mathbf{X}, \sigma, \lambda) &= p(\beta \mid \mathbf{X}, \sigma, \lambda) p(\mathbf{y} \mid \mathbf{X}, \beta, \sigma, \lambda) \\
 &= p(\beta \mid \lambda) p(\mathbf{y} \mid \mathbf{X}, \beta, \sigma) \\
 &= \underbrace{\mathcal{N}(\beta \mid \mathbf{0}, \lambda \mathbf{I})}_{\text{prior}} \underbrace{\mathcal{N}(y_n \mid \beta^T \mathbf{x}_n, \sigma^2)}_{\text{likelihood}}. \tag{A.7}
 \end{aligned}$$

The first line in (A.7) uses the product rule of probability. The second line follows from the dependency structure in M_1 , as seen in the *PGM* in Figure A.2a. The third line follows from the *generative story* we chose for M_1 , as in Figure A.1 on the preceding page.

Equation (A.7) thus demonstrates several aspects common to all MBML models. We see that the three key components of MBML – PGM, generative story, and joint probability distribution – together yield the factorization of a centrally important quantity. This factorization is a *product of conditionally independent* factors, which reflects the relationship between distributions as dictated by

Bayes' rule:

$$\text{posterior} \propto \text{prior} \times \text{likelihood}, \quad (\text{A.8})$$

where $\propto$ indicates that both sides are equal up to a multiplicative constant.

To recap, in model M_1 of taxi pickups, the posterior distribution of the latent linear coefficients $\beta = (\beta_1, \dots, \beta_L)^T$ follows from the prior and likelihood. We have defined the prior on β as a Gaussian, and the likelihood of the observations as another Gaussian, which is parameterized on β . The analytical forms of the prior and likelihood appear in (A.7), which provides the *joint* distribution of parameters β *together* with observations $\mathbf{y}$. However, we are now interested in the *marginal* posterior distribution of *only* β , as we already know $\mathbf{y}$. Next, we describe methods for learning the latent parameters of any probabilistic model.

A.2.3 Inference

To learn the relationship between $\mathbf{y}$ and $\mathbf{X}$, we have formulated linear regression model M_1 , through which we wish to learn coefficients β . On one hand, we could use the *Frequentist approach*, whereby we obtain only point estimates for the parameters, e.g. the value of β which maximizes some probability of interest. For example, we could compute a Maximum Likelihood Estimate (MLE), which maximizes (A.4), or a Maximum a-posteriori Estimate (MAP), which maximizes (A.7). Deterministic algorithms for computing MLE and MAP are described in e.g. (Murphy, 2012).

On the other hand, point estimates have some drawbacks, e.g. MLE can lead to overfitting, because MLE is calculated to best fit the observed data. More generally, point estimates do not conserve *uncertainty* about model parameters. In contrast, we can retain a good measure of uncertainty about model parameters, if we infer their posterior *distribution*. E.g. for any value of β in M_1 , the posterior distribution $p(\beta | \mathbf{y})$ tells us how likely it is that the linear coefficients have that value, given the observed taxi pickups $\mathbf{y}$. Hence in MBML, our goal is to perform *inference*: obtain the *posterior distribution* of model parameters.

A.2.3.1 Bayesian Inference

Bayesian Inference is a systematic way for inferring the posterior distribution of model parameters, given the prior and likelihood distributions, through *Bayes'*

Figure A.3: An intractable posterior distribution (left) can be approximated either by sampling (middle), or by optimizing a tractable proxy distribution (right).

rule:

$$\overbrace{p(\beta | \mathbf{y})}^{\text{posterior}} = \frac{\overbrace{p(\beta)}^{\text{prior}} \overbrace{p(\mathbf{y} | \beta)}^{\text{likelihood}}}{\underbrace{p(\mathbf{y})}_{\text{evidence}}} . \quad (\text{A.9})$$

For any value of β , the numerator in (A.9) is typically easy to compute, because it consists of distributions which we explicitly selected. E.g. for model M_1 , (A.7) defines this numerator in closed and tractable form. The evidence distribution $p(\mathbf{y})$ in the denominator is a *normalization constant*, equal to the sum (or integral) of the numerator over all possible values of β . As such, the normalization constant is not necessarily tractable to compute, e.g. if its analytic form cannot be formulated, or if there are too many possible values of β for direct summation.

A.2.3.2 Exact vs. Approximate Bayesian Inference

When the normalization constant in (A.9) is tractable, we can infer the desired left-hand side precisely through several *exact Bayesian inference* methods, see e.g. (Jensen, 1996). Because the prior and likelihood in M_1 are multivariate Gaussians, the normalization constant for M_1 is analytically tractable through properties of the Normal distribution, as discussed in Section 2.3 of (Bishop, 2006).

Generally though, the denominator in (A.9) is often intractable, in which case we can only obtain an approximation of the desired left-hand side. One way to perform *approximate Bayesian inference* is to collect *samples* from the distribution of the tractable numerator, and use the samples to approximate quantities related to the posterior distribution (Gilks, 2005). Another way is to search for an analytically tractable *proxy distribution* for the true posterior distribution (Jordan et al., 1999). Figure A.3 illustrates both manners of approximate inference.

```

data { // What is already observed. This will be given as input.
  int<lower=0> N; // Number of feature vectors.
  int<lower=0> D; // Number of features.
  matrix[N, D] X; // Feature vectors.
  vector[N] y; // Target variables.
}
parameters { // What we wish Stan to infer.
  vector[D] beta; // Linear coefficients for mean of y.
  real sigma; // Standard deviation of y.
}
model { // Priors and likelihood.
  for (i in 1:D)
    beta[i] ~ normal(0, 10); // Prior.
  sigma ~ cauchy(0, 1); // Prior.
  y ~ normal(X * beta, sigma); // Likelihood (mean is vector[N]).
}

```

Listing A.1: Stan program for model M_1 , with std. dev. σ as a latent variable rather than pre-fixed, and $\lambda = 10$. The choice of Cauchy prior is per (Gelman et al., 2008).

There exist multiple probabilistic programming tools which support approximate Bayesian inference (Gordon et al., 2014). One such popular tool is *Stan* (Team, 2018), and along this chapter, we provide Stan programs for some of the models we build. The reader may notice that each Stan program closely resembles the generative process for the model. Listing A.1 presents a Stan program for a model similar to M_1 , except that σ is latent instead of known in advance.

A.2.4 Model Improvements

In our initial model M_1 , the target variable $\mathbf{y}$ corresponds to taxi pickups, and is normally distributed as $\mathbf{y} \sim \mathcal{N}(\mathbf{y} \mid \beta^T \mathbf{X}, \sigma^2)$. Let us now present two separate model improvements, by enhancing the model with some domain-specific knowledge about our case study. These improvements also demonstrate the flexibility of the MBML approach, as the corresponding model changes are straightforward to apply.

A.2.4.1 Heteroscedasticity

Upon further inspection, we detect that the number of taxi pickups varies more wildly at some time intervals than others, e.g. there are nights which exhibit a far wider range of taxi demand than usual. As such, the time series of pickups is *heteroscedastic*, namely its variance changes over time.

To account for heteroscedasticity, we shall model the noise as input-dependent.

Given $\mathbf{X}, \tau, \lambda$:

1. Draw $\beta \sim \mathcal{N}(\beta \mid \mathbf{0}, \lambda \mathbf{I})$.
2. Draw $\eta \sim \mathcal{N}(\eta \mid \mathbf{0}, \tau \mathbf{I})$.
3. For each feature vector $\mathbf{x}_n, n = 1 \dots N$:
 - 3.1. Draw target $y_n \sim \mathcal{N}(y_n \mid \beta^T \mathbf{x}_n, \exp \{\eta^T \mathbf{x}_n\})$

Figure A.4: Generative process for heteroscedastic model M_1^{Het}

We could e.g. model $\varepsilon_n \sim \mathcal{N}(\varepsilon_n \mid 0, \eta^T \mathbf{x}_n)$ for each $n = 1..N$, thus replacing the constant variance σ^2 with $\eta^T \mathbf{x}_n$, where η is another vector of coefficients to be inferred. However, $\eta^T \mathbf{x}_n$ might be negative for some $\mathbf{x}_n$, whereas variance is never negative. So instead, we model the noise as following.

$$\varepsilon_n \sim \mathcal{N}(\varepsilon_n \mid 0, \exp \{\eta^T \mathbf{x}_n\}) . \quad (\text{A.10})$$

The exponent in (A.10) ensures that the variance is positive at all times, as $\exp: \mathbb{R} \rightarrow (0, +\infty)$. This is an example of using a *link function*, which in this case is the *log link*, because link functions are typically named after the inverse transformation. Equivalently, in the new heteroscedastic model M_1^{Het} , the target variables are distributed as

$$y_n \sim \mathcal{N}(y_n \mid \beta^T \mathbf{x}_n, \exp \{\eta^T \mathbf{x}_n\}) . \quad (\text{A.11})$$

Similarly to β , our prior belief for η is a multivariate Gaussian, parameterized by some pre-fixed hyper-parameter τ . The corresponding generative process for M_1^{Het} is in Figure A.4.

Figure A.2b on page 173 provides the PGM for M_1^{Het} . The corresponding joint probability distribution of $\mathbf{y}, \beta, \eta$ thus factorizes as

$$p(\mathbf{y}, \beta, \eta \mid \mathbf{X}, \tau, \lambda) = \underbrace{\mathcal{N}(\beta \mid \mathbf{0}, \lambda \mathbf{I}) \mathcal{N}(\eta \mid \mathbf{0}, \tau \mathbf{I})}_{\text{priors}} \underbrace{\mathcal{N}(y_n \mid \beta^T \mathbf{x}_n, \exp \{\eta^T \mathbf{x}_n\})}_{\text{likelihood}} . \quad (\text{A.12})$$

Listing A.2 on the following page provides a Stan program for M_1^{Het} .

To compare the *predictive performance* of M_1 and M_1^{Het} , we first partition the data $\mathbf{X}$ and the observations $\mathbf{y}$ into two parts: a “*train set*” ($\mathbf{X}^{\text{train}}, \mathbf{y}^{\text{train}}$), and a “*test set*” ($\mathbf{X}^{\text{test}}, \mathbf{y}^{\text{test}}$). $\mathbf{y}^{\text{train}}$ consists of 5781 randomly chosen observations (approx. 2/3 of $\mathbf{y}$), and $\mathbf{y}^{\text{test}}$ consists of the remaining 2979 observations (approx. 1/3 of $\mathbf{y}$). Data vectors $\mathbf{X}^{\text{train}}, \mathbf{X}^{\text{test}}$ are the rows in $\mathbf{X}$ which correspond respectively to $\mathbf{y}^{\text{train}}, \mathbf{y}^{\text{test}}$. We then fit each of M_1, M_1^{Het} on the train set, and run the trained models on $\mathbf{X}^{\text{test}}$ to obtain predictions.

```

data { ... // Same as for the initial model. }
parameters {
  vector[D] beta;
  vector[D] eta;
}
model {
  beta ~ cauchy(0, 10); // Vectorized.
  eta ~ cauchy(0, 10);
  y ~ normal(X * beta, exp(X * eta)); // Heteroscedastic likelihood.
}

```

Listing A.2: Stan program for model M_1^{Het} with $\tau = 10$. Differences from the initial model (Listing A.1) are highlighted.

Figure A.5 on the next page illustrates the predictions of M_1 and M_1^{Het} for a randomly selected subset of the test set. For each model, we plot a confidence interval as ± 1 standard deviation around the mean prediction. While the confidence intervals of both models capture most of the true observations, the varying std. dev. of M_1^{Het} is often lower than the fixed std. dev. of M_1 . Hence in this case study, the heteroscedastic model is often more confident than the homoscedastic model, while maintaining comparable accuracy. Finally, we see that both models can yield negative predictions, and address this issue in the next section.

A.2.4.2 Count Data

In our initial model M_1 , the likelihood $\mathcal{N}(y_n | \beta^T \mathbf{x}_n, \sigma^2)$ is continuous, and can yield negative, non-integer draws. However, each target variable y_n in this case is actually a discrete count of taxi pickups, namely a non-negative integer. We can thus obtain a better model by modeling y through some discrete distribution for *count data*, such as the Poisson distribution.

The *Poisson distribution* pertains to independent counts of occurrences within successive spatio-temporal intervals of fixed size, where the rate of occurrences is a known constant. We have already assumed that successive counts of taxi pickups are independent. However, a closer look at the data reveals that the mean number of pickups is unstable over time. We therefore choose to model $y_1, \dots, y_N$ as differently distributed, so that each y_n is Poisson distributed according to the data at time n , namely,

$$y_n \sim \text{Poisson}(y_n | \exp\{\beta^T \mathbf{x}_n\}), \quad (\text{A.13})$$

where the exponent again ensures that no rate is negative. We have thus obtained a more appropriate model M_1^{Pois} , which has a generative process as in Figure A.6 on the facing page.

Figure A.5: Predictive performance of homoscedastic M_1 vs. heteroscedastic M_1^{Het} , for 48 randomly chosen observations from the test set. Mean prediction $\hat{\mu}$ is nearly the same for both models, and is thus plotted only once. $\hat{\sigma}_1$ and $\hat{\sigma}_1^{\text{Het}}$ are the prediction standard deviations of M_1 and M_1^{Het} , respectively.

Given $\mathbf{X}, \lambda$:

1. Draw $\beta \sim \mathcal{N}(\beta \mid \mathbf{0}, \lambda \mathbf{I})$.
2. For each feature vector $\mathbf{x}_n$, $n = 1 \dots N$:
 - 2.1. Draw target $y_n \sim \text{Poisson}(y_n \mid \exp\{\beta^T \mathbf{x}_n\})$

Figure A.6: Generative process for M_1^{Pois}

The PGM for M_1^{Pois} appears in Figure A.2c on page 173, and is obtained by omitting the now unused hyper-parameter σ from the PGM for M_1 . Consequently, the joint probability distribution of β and $\mathbf{y}$ in M_1^{Pois} factorizes as following.

$$p(\beta, \mathbf{y} \mid \mathbf{X}, \sigma, \lambda) = \underbrace{\mathcal{N}(\beta \mid \mathbf{0}, \lambda \mathbf{I})}_{\text{prior}} \underbrace{\prod_{n=1}^N \text{Poisson}(y_n \mid \exp\{\beta^T \mathbf{x}_n\})}_{\text{likelihood}}, \quad (\text{A.14})$$

where

$$\text{Poisson}(y_n \mid \exp\{\beta^T \mathbf{x}_n\}) = \frac{\exp\{y_n \beta^T \mathbf{x}_n - \exp(\beta^T \mathbf{x}_n)\}}{y_n!} \quad (\text{A.15})$$

is the value of the Poisson PDF. We can derive the posterior distribution of M_1^{Pois} only through approximate inference, because (A.14) yields an intractable

```
data { ... // Same as for the initial model. }
parameters { // Unused parameter sigma is omitted.
  vector[D] beta;
}
model {
  beta ~ cauchy(0, 10);
  y ~ poisson_log(X * beta); // Poisson likelihood, log link function.
}
```

Listing A.3: Stan program for M_1^{Pois} . Differences from initial model M_1 (Listing A.1) are highlighted.

Statistic	Formula	Units	Comments
Mean Absolute Error (MAE)	$\frac{1}{V} \sum_{v=1}^V y_v^{test} - \hat{y}_v $	Same as observations	Lower is better, 0 is perfect fit.
Rooted Mean Squared Error (RMSE)	$\sqrt{\frac{1}{V} \sum_{v=1}^V (y_v^{test} - \hat{y}_v)^2}$	Same as observations	Lower is better, 0 is perfect fit.
Coefficient of Determination (R^2)	$1 - \frac{(y_v^{test} - \hat{y}_v)^2}{(y_v^{test} - \bar{y}^{test})^2}$	Unitless	Higher is better, 1 is perfect fit.

Table A.2: Summary statistics for prediction errors, where $y_1^{test}, \dots, y_V^{test}$ are V test observations, $\hat{y}_1, \dots, \hat{y}_V$ are the corresponding predictions, and $\bar{y}^{test} = (y_1^{test}, \dots, y_V^{test}) / V$ is the mean of the test observations.

normalization constant in Bayes’ rule (A.9). A Stan program for M_1^{Pois} appears in Listing A.3.

Finally, let us compare the *predictive performance* of M_1^{Pois} and M_1 , using the same train and test sets as we defined earlier. This time, we measure performance through some commonly used summary statistics of prediction error (Li, 2017), as defined in Table A.2. The results appear in Table A.3, which shows that Poisson regression outperforms linear regression in this case study.

Model	MAE	RMSE	R^2
M_1	19.292	25.879	0.645
M_1^{Pois}	18.128	24.711	0.676

Table A.3: Predictive performance of M_1^{Pois} vs. M_1 . The best value in each column is highlighted in bold.

A.3 Case Study 2: Travel Mode Choices

We have concluded the first case study by modeling the target variable as drawn from a discrete distribution. Let us now examine another case study, in which the target variable is discrete, although categorical rather than numeric. Here, we wish to model *discrete choice of travel mode*, which takes one of distinct values: bus, train, car, or plane.

The dataset is a travel diary, which consists of $N = 394$ trips by $G = 80$ different individuals. For each trip $n = 1..N$, the data features $\mathbf{x}_n$ are properties of the trip – e.g. cost and duration – and possibly also demographic information about the individual g_n who took the trip. The target variable y_n for each trip is the travel mode, which g_n chose among the $K = 4$ abovementioned options.

Our goal is to model how each individual chooses the travel mode for a trip, given the trip properties. Such a model has many possible applications, e.g. understanding of human behavior, planning pricing policies, or incentivizing the usage of “green” transportation. Because the target variable in this model takes one of a finite number of categories (“classes”), this case study is an example of a *classification problem*.

As in the previous case study, our initial model M_2 is a *Generalized Linear Model* (GLM), i.e. at the core of M_2 are linear combinations of the data features. We begin by numbering the travel modes (“classes”) arbitrarily as $1..K$, and assign for each class $k = 1..K$ a parameter vector β_k with as many elements as there are data features. Our goal is thus to infer the posterior distribution of model parameters $\beta_1, \dots, \beta_K$.

We finish building M_2 by modeling each y_n as a draw from a *categorical distribution* (Cat). That is, each class $k = 1..K$ has some probability $p_n^{(k)}$ of being drawn as the n ’th observation. To model these probabilities, we use *logistic regression*, so that for all $k = 1..K$:

$$\begin{aligned} p_n^{(k)} &= (\text{Softmax}(\beta_1^T \mathbf{x}_n, \dots, \beta_K^T \mathbf{x}_n))_k \\ &\triangleq \frac{\exp(\beta_k^T \mathbf{x}_n)}{\exp(\beta_1^T \mathbf{x}_n) + \dots + \exp(\beta_K^T \mathbf{x}_n)}. \end{aligned} \quad (\text{A.16})$$

Hence for model M_2 , the generative process is as in Figure A.7a on the next page, where λ is a pre-fixed hyper-parameter. The corresponding PGM for M_2 appears in Figure A.7b on the following page.

Figure A.7: Classification model M_2 : (a) generative process, (b) Probabilistic Graphical Model.

```

data {
  int N, D, K; // No. observations, features, classes.
  matrix[N, D] X; // Input features.
  int y[N]; // Output classes.
}
parameters {
  matrix[K, D] beta; // Feature coefficients for each class.
}
model {
  for (k in 1:K) {
    for (d in 1:D) {
      beta[k, d] ~ normal(0, 10); // Prior.
    }
  }
  for (i in 1:N) {
    y[i] ~ categorical(softmax(beta * X[i, :]')); // Likelihood.
  }
}

```

Listing A.4: Stan program for classification model M_2 .

The joint probability distribution for M_2 thus factorizes as

$$\begin{aligned}
 p(\mathbf{y}, \beta_1, \dots, \beta_K \mid \mathbf{X}, \lambda) &= \underbrace{\left(\prod_{k=1}^K \mathcal{N}(\beta_k \mid \mathbf{0}, \lambda \mathbf{I}) \right)}_{\text{priors}} \times \\
 &\quad \underbrace{\prod_{n=1}^N \text{Cat}\{y_n \mid \text{Softmax}(\beta_1^T \mathbf{x}_n, \dots, \beta_K^T \mathbf{x}_n)\}}_{\text{likelihood}}. \quad (\text{A.17})
 \end{aligned}$$

For any value of $\beta_1, \dots, \beta_K$, the posterior probability in M_2 can be inferred from (A.17) only approximately, because the normalization constant in Bayes' rule (A.9) is intractable. Listing A.4 shows a Stan program for M_2 .

A.3.1 Improvement: Hierarchical Modeling

Our initial model M_2 has one set of parameters β_k for each class (travel mode) $k = 1..K$. In other words, M_2 is strongly “pooled”, as all observations for the same class share the same parameters. We will now revise this modeling assumption, and improve the model by constructing a *hierarchy of parameters* in two successive steps.

First, we note that for any class k , β_k is shared among all G individuals, as also seen in the PGM in Figure A.7b on the facing page. However, different individuals could surely have very different travel mode preferences. Therefore, as a first step, we change the model to have one parameter vector $\beta_k^{(g)}$ for each class k and each individual $g = 1..G$. This has the same effect as grouping the trips of each individual, and assigning each group its own separate parameters. The resulting, intermediate model M_2^{Hier1} appears in Figure A.8a on the next page.

However, this change alone will likely lead to terrible overfitting in our case, because the dataset has many more features than observations per group (i.e. per individual). We counter this by also modeling a restriction, whereby for each class k , all parameters $\beta_k^{(1)}, \dots, \beta_k^{(G)}$ are drawn from the same distribution. In this manner, our prior belief is that travel mode preferences can differ between individuals, but are nevertheless drawn from a common distribution. Our new and improved model M_2^{Hier2} thus comprises of a two-level hierarchy of parameters, as seen in the PGM in Figure A.8b on the following page.

For each class $k = 1..K$, we let the common distribution of parameters $\beta_k^{(1)}, \dots, \beta_k^{(G)}$ in M_2^{Hier2} be a multivariate Gaussian with mean μ_k and positive variance $\exp(\sigma_k)$, so that μ_k and σ_k are additional latent parameters. Figure A.9 on page 185 gives the generative process for M_2^{Hier2} , while Listing A.5 on page 186 provides a corresponding Stan program.

From here on, we stop specifying how the joint probability distribution factorizes for each model. As before, this factorization can be derived directly from the PGM and generative process for the model.

Hierarchical modeling is thus a compromise between two extremes. One extreme is too much pooling, such as assigning the same parameters to all groups (Figure A.7), while the other extreme is too little pooling, such as assigning separate parameters for each group (Figure A.8a). When modeling in general, one chooses the degree of pooling based on the data and the assumed priors.

Figure A.8: Probabilistic Graphical Models for case study 2: (a) intermediate, one-level hierarchical model M_2^{Hier1} , (b) final, two-level hierarchical model M_2^{Hier2} . Different colors indicate different groups.

A.4 Case Study 3: Freeway Occupancy in San Francisco

Our third case study deals with a *temporal dataset*: lane occupancy rates in a freeway in San Francisco, as measured by a sensor on the road. The measurements are aggregated into consecutive 10 minute intervals, so that the resulting time series $\mathbf{y} = y_1, \dots, y_T$ consists of $T = 1008$ consecutive real values (“lags”), each between 0 and 1.

Our goal is to model the freeway occupancy rates. Some applications of such a model include: prediction of future occupancy rates for better traffic routing, real-time detection of extraordinary queuing, and better understanding of driver behavior. To this end, we next build and incrementally improve a model, as well as introduce the highly versatile families of State-Space Models (SSM) and Linear Dynamical Systems (LDS).

A.4.1 Autoregressive Model

Our first model M_3 is $\text{AR}(k)$, namely an *autoregressive* model of some pre-fixed order k . In M_3 , each lag y_t is a linear combination of the k preceding lags, plus white noise. Thus in the PGM for M_3 , the nodes are $y_1, \dots, y_T$, and the edges are from each y_n to each of $y_{n+1}, \dots, y_{n+k}$. The generative process for M_3 appears in Figure A.11 on page 187; note the special treatment of the first

Given $\mathbf{X}, K, g_1, \dots, g_N, \lambda, \tau$:

1. For each class $k = 1, \dots, K$:
 - 1.1. Draw global mean $\mu_k \sim \mathcal{N}(\mu_k \mid \mathbf{0}, \lambda \mathbf{I})$.
 - 1.2. Draw global std. dev. $\sigma_k \sim \mathcal{N}(\sigma_k \mid 0, \tau)$.
 - 1.3. For each group $g = 1, \dots, G$:
 - 1.3.1. Draw coefficients $\beta_k^{(g)} \sim \mathcal{N}(\beta_k^{(g)} \mid \mu_k, \exp\{\sigma_k\} \mathbf{I})$.
2. For each $n = 1, \dots, N$:
 - 2.1. Draw class $y_n \sim \text{Cat}\left(y_n \mid \text{Softmax}\left\{\left(\beta_1^{(g_n)}\right)^T \mathbf{x}_n, \dots, \left(\beta_K^{(g_n)}\right)^T \mathbf{x}_n\right\}\right)$.

Figure A.9: Generative process for model M_2^{Hier2}

Figure A.10: PGM for model M_2^{Hier2} – a concise equivalent of Figure A.8b.

k observations, for which some previous lags are unknown.

A.4.2 State-Space Model

Noise in M_3 may build up over time, because each observation depends on k previous observations, themselves noisy. We thus move to a more robust, *State-Space Model* (SSM) M_3^{SSM} , in which the observations are generated (“emitted”) by a stochastic state. That is, M_3^{SSM} assumes that occupancy rates evolve over time through some latent, non-deterministic process, which is visible only through its observed measurements $y_1, \dots, y_T$.

As such, M_3^{SSM} has a set of *hidden state* variables $\mathbf{h}_1, \dots, \mathbf{h}_T$: vectors of dimension D , which we are free to choose. Each state yields the next state per some *transition distribution*. Each observation y_t is then a draw from a *measurement distribution*, which depends on the corresponding hidden state $\mathbf{h}_t$. Figure A.12a on page 187 shows the basic structure of a PGM for M_3^{SSM} . Special treatment is required only for the first hidden state $\mathbf{h}_1$, for which no preceding hidden state is defined.

```

data {
  int N, D, K;   matrix[N, D] X;   int y[N]; // As in model M_2.
  real<lower=0> lambda, tau; // Non-negative hyper-parameters.
  int G; // Number of groups.
  int g[N]; // Group label for each x_n, y_n.
}
parameters {
  matrix[K, G] beta; // Feature coefficients for each group.
  vector[K] mu; // For shared mean prior per class.
  vector[K] sigma; // For shared std. dev. prior per class.
}
model {
  for (k in 1:K) { // Priors.
    mu[k] ~ normal(0, lambda);
    sigma[k] ~ normal(0, tau);
    // 2nd dimension (G) is vectorized:
    beta[k] ~ normal(mu[k], exp(sigma[k]));
  }
  for (n in 1:N) { // Likelihood.
    y[n] ~ categorical(softmax(beta[:, g[n]] * X[n, :]' ));
  }
}

```

Listing A.5: Stan program for hierarchical model M_2^{Hier2} . Note the difference in the 2nd dimension of beta vs. the Stan program for the completely pooled model M_2 (Listing A.4).

A.4.3 Linear Dynamical Systems

It remains to describe the form of the transition and measurement distributions. For computational tractability, we assume these distributions are linear Gaussians, so that

$$\mathbf{h}_t \sim \mathcal{N}(\mathbf{h}_t \mid \mathbf{B}\mathbf{h}_{t-1}, \mathbf{R}), \quad (\text{A.18})$$

$$y_t \sim \mathcal{N}(y_t \mid \mathbf{c}^T \mathbf{h}_t, \sigma^2), \quad (\text{A.19})$$

where $\mathbf{B}^{D \times D}$, $\mathbf{R}^{D \times D}$, $\mathbf{c}^{D \times 1}$, σ are latent model parameters to be inferred. The corresponding PGM appears in Figure A.12b on the next page. This model, which we denote as M_3^{LDS} , is called a *Linear Dynamical System* (LDS). It is also widely known as *Linear Kalman Filter* in control theory literature (Grewal, 2011).

For any choice of priors for the latent parameters $\mathbf{B}, \mathbf{R}, \mathbf{c}, \sigma$, we can obtain the generative process for M_3^{LDS} , by combining the PGM in Figure A.12b with distributions (A.18) and (A.19). By imposing *structural restrictions* on parameter matrices $\mathbf{B}, \mathbf{R}$ in M_3^{LDS} , many popular time series can be modeled (including $AR(k)$), and inference becomes less computationally demanding (Nasrabadi, 2007).

Given $\mathbf{y}, k, \lambda, \sigma, \mu_0$:

1. Draw transition coefficients $\beta = (\beta_1, \dots, \beta_k) \sim \mathcal{N}(\beta \mid \mathbf{0}, \lambda \mathbf{I})$.
2. Draw first observation $y_1 \sim \mathcal{N}(y_1 \mid \mu_0, \sigma^2)$.
3. For $i = 2, \dots, k$:
 - 3.1. Draw observation $y_i \sim \mathcal{N}(y_i \mid \beta_1 y_{i-1} + \beta_2 y_{i-2} + \dots + \beta_{i-1} y_1, \sigma^2)$.
4. For $t = (k + 1), \dots, T$:
 - 4.1. Draw observation $y_t \sim \mathcal{N}(y_t \mid \beta_1 y_{t-1} + \beta_2 y_{t-2} + \dots + \beta_k y_{t-k}, \sigma^2)$.

Figure A.11: Generative process for $AR(k)$ model. Steps 2 and 3 handle the first k observations, for which some previous lags are unknown. Step 4 handles the subsequent observations, for which all k previous lags are known.

Figure A.12: PGMs for (a) general State-Space Model (SSM), and (b) Linear Dynamical System (LDS), a sub-class of SSM.

A.4.4 Common Enhancements to LDS

Let us now show that LDS models are easily adaptable to various common usage scenarios. Some of the PGMs for the enhanced models may appear very similar to the PGM of M_3^{LDS} in Figure A.12b, but the underlying probabilistic models are significantly different nonetheless.

A.4.4.1 Filling Gaps

Suppose that the time series has some missing lags in-between 1 to T . We can perform *imputation* for any missing value y_t , simply by modeling y_t as a latent (i.e. unobserved) measurement from state $\mathbf{h}_t$. Similarly, to generate *predictive distributions* for $r \geq 1$ lags immediately after T , we simply extend

Figure A.13: PGM for an LDS model with both imputation at time step 2 and forecasting of time step $T + 1$.

M_3^{LDS} to include latent states $\mathbf{h}_{T+1}, \dots, \mathbf{h}_{T+r}$, along with corresponding latent measurements $y_{T+1}, \dots, y_{T+r}$. Figure A.13 illustrates both changes to the PGM of M_3^{LDS} .

Following is an example of data imputation for this case study of lane occupancy rates. First, we randomly select 144 observations in $\mathbf{y}$, then partition the selection randomly into a train set $\mathbf{y}^{\text{known}}$ of size 29, and a test set $\mathbf{y}^{\text{missing}}$ of size 115. As nearly 80% of observations are rendered missing, this is a relatively difficult imputation setting. We then fit M_3^{LDS} on $\mathbf{y}^{\text{known}}$, and run the trained model on $\mathbf{y}^{\text{missing}}$. Figure A.14 on the facing page illustrates the prediction quality, and shows that M_3^{LDS} fills in the missing data rather accurately.

A.4.4.2 External Data

To take advantage of any additional data $\mathbf{x}_1, \dots, \mathbf{x}_T$, e.g. weather or seasonality information, we can augment M_3^{LDS} simply as

$$\mathbf{h}_t \sim \mathcal{N}(\mathbf{h}_t \mid \mathbf{B}\mathbf{h}_{t-1} + \mathbf{W}\mathbf{x}_t, \mathbf{R}), \quad (\text{A.20})$$

where $\mathbf{W}$ is an additional matrix of latent coefficients to be inferred. Figure A.15a on page 190 shows one time step of the PGM for the augmented model.

A.4.4.3 Regimes

Suppose now that occupancy rates behave differently under different *regimes*, such as congested traffic, free flow, road incidents, or harsh weather. We can

Figure A.14: Occupancy rate imputation with M_3^{LDS} for 144 randomly selected observations, 80% of which are treated as missing.

model such regimes with a *Switching LDS*, the PGM of which appears in Figure A.15b on the following page, as following.

First, we pre-fix the number of regimes M , and assign to each regime $m = 1..M$ a dedicated chain of latent states $\mathbf{h}_1^{(m)}, \dots, \mathbf{h}_T^{(m)}$. Then, at time step $t = 1..T$, we let categorical variable z_t choose which of regimes $1..M$ applies. Finally, we model y_t as a measurement from $\mathbf{h}_t^{(z_t)}$: the hidden state corresponding to the regime which z_t chooses. We can also assign (perhaps hierarchically) matrices $\mathbf{B}^{(m)}, \mathbf{R}^{(m)}$ for the transition distribution of chain m , because different regimes can evolve differently over time.

A.4.5 Non-Linear Variations on LDS

We conclude the discussion of LDS by introducing two commonly used variations, wherein the transition and measurement distributions incorporate *non-linear functions*. One popular variation is the *Extended Kalman Filter* (EKF), which replaces the linear combinations in (A.18) and (A.19) with freely chosen, *differentiable functions* f and g , namely,

$$\mathbf{h}_t \sim \mathcal{N}(\mathbf{h}_t \mid f(\mathbf{h}_{t-1}), \mathbf{R}), \quad (\text{A.21})$$

$$y_t \sim \mathcal{N}(y_t \mid g(\mathbf{h}_t), \sigma^2). \quad (\text{A.22})$$

EKF is at the foundation of a very wide range of applications, including many navigation systems (Wan, 2006).

Figure A.15: One time step t in the PGM for an enhanced LDS model, with changes to M_3^{LDS} highlighted in color: (a) external data, (b) M regimes with separate transition distributions.

Another popular variation on LDS is the *Hidden Markov Model* (HMM) (Schuster-Böckler and Bateman, 2007), which uses discrete, categorical distributions for the transitions and measurements. Without loss of generality, let us assume the same set of possible values $1..K$ for both hidden states and measurements.

In HMM, the transition distribution is specified as a matrix $\mathbf{B} \in \mathbb{R}^{K \times K}$. For all $i, j = 1..K$, $\mathbf{B}_{i,j}$ is the probability to obtain state value j , given that the preceding state had value i . Matrix $\mathbf{C} \in \mathbb{R}^{K \times K}$ similarly specifies the measurement distribution. Hence in HMM, matrices $\mathbf{B}$ and $\mathbf{C}$ are latent parameters, which we intend to infer. Figure A.16 on the next page presents the corresponding PGM.

For any hidden state value, the corresponding rows in $\mathbf{B}$ and $\mathbf{C}$ cover *all possible* transitions and measurements from that state, respectively. Therefore, each row in $\mathbf{B}$ and in $\mathbf{C}$ is a vector of k probabilities, which sum to 1. We can draw such normalized probability vectors from Dir, the continuous *Dirichlet distribution* ((Bishop, 2006) Section 2.2.1), for which we use hyper-parameters $\alpha, \beta \in \mathbb{R}^k$, as in the generative process in Figure A.17 on the facing page.

Finally, we remark that it is also common to use hybrid HMM-LDS models, which combine e.g. discrete hidden states with continuously distributed measurements (Rabiner, 1989).

Figure A.16: PGM for Hidden Markov Model (HMM).

Given $\mathbf{y}, K, \alpha \in \mathbb{R}^K, \beta \in \mathbb{R}^K$:

1. For each $k = 1, \dots, K$:
 - 1.1. Draw transition probabilities $\mathbf{b} \sim \text{Dir}(\mathbf{b} \mid \alpha)$.
 - 1.2. Draw measurement probabilities $\mathbf{c}_k \sim \text{Dir}(\mathbf{c}_k \mid \beta)$.
2. For each time step $t = 1, \dots, T$:
 - 2.1. Draw hidden state $\mathbf{h}_t \sim \text{Cat}(\mathbf{h}_t \mid \mathbf{b}_{\mathbf{h}_{t-1}})$.
 - 2.2. Draw observed measurement $y_t \sim \text{Cat}(y_t \mid \mathbf{c}_{\mathbf{h}_t})$.

Figure A.17: Generative process for Hidden Markov Model. $\mathbf{B}_k$ and $\mathbf{c}_k$ denote the k 'th row in $\mathbf{B}$ and $\mathbf{C}$, respectively.

A.5 Case Study 4: Incident Duration Prediction

Textual data is often an additional source of useful information for statistical models, e.g. websites and social media can provide prediction models with highly pertinent context. However, textual data is non-numeric, often unstructured, and poses challenges of processing natural language.

This case study is based on (Pereira et al., 2013), and gives a methodological example of combining textual and numeric data in a prediction model. The dataset consists of police reports of $D = 10000$ incidents in years 2010-2011 in several expressways. For each report, we have *numeric features* – such as incident time, number of affected lanes, and duration until clearance – and a *textual document*, which describes the incident chronology in free-form text.

Our goal is to build a model for predicting incident duration. Such a model

Figure A.18: Prediction of incident duration in minutes, either without (a) or with (b) text-based features. The blue line indicates ideal performance, while the red dashed lines correspond to linear fits on the observations. The plot for each model also shows the percentage of prediction errors which are smaller than 10 minutes or 20 minutes. (Pereira et al., 2013)

is also useful for studying the impact of incidents on traffic. We begin by performing regression on only the numeric features, as described in (Pereira et al., 2013). Figure A.18a shows that the predictions of this initial model M_4^{NoText} are far from ideal.

For the rest of this case study, we describe how to *encode* each textual document compactly as a numeric vector. We can then concatenate the text-based encodings with the corresponding numeric features, to obtain unified data vectors for all the documents.

Finally, we fit a regression model M_4^{WithText} on the unified data vectors, and obtain significantly better predictive performance than M_4^{NoText} . As Figure A.18b shows for M_4^{WithText} , the predictions are much closer to ideal, and there are significantly less observations for which the prediction error is greater than 10 (or 20) minutes.

A.5.1 Preprocessing

As a first common step in dealing with textual information, we preprocess each document into a simplified textual form, which is easier to encode. For instance, we reduce each word to its stem (e.g. each of “injury”, “injuries”, and “injured” becomes “injur”), and remove common function words (e.g. “the”, “is”, and “which”). We then collect all unique words in the processed documents into an

Figure A.19: (a) Example incident report, before preprocessing. (b) Bag-of-Words encoding of the report. (c) Much shorter encoding of the report through topic modeling. (d) Example topic, which pertains mainly to serious injuries. (Pereira et al., 2013)

arbitrarily ordered list $C = (C_1, C_2, \dots)$, which we call “the *dictionary*”. In this case study, the length $|C|$ of the dictionary is in the thousands.

A.5.2 Bag-of-Words Encoding

Any preprocessed document can now be encoded numerically as a *Bag of Words* (BoW): a vector where the j ’th element is the frequency of dictionary word C_j in the document. It may thus be tempting to encode each of the processed documents as BoW, and feed the BoW vectors along with the other numeric features into a prediction model. However, doing so could greatly hinder learning, because the dimension $|C|$ of a BoW vector is typically large, as in this case study. We next overcome this problem through Topic Modeling, which yields for each document a numeric encoding much shorter than BoW, as illustrated in Figure A.19.

A.5.3 Latent Dirichlet Allocation (LDA)

Going over the incident reports, we notice that they seem to revolve around several repeated topics, such as: incident severity, casualties, oil spill, road blockage, and emergency services. Each textual report thus pertains to a mixture of topics of different proportions, although it is uncertain what the topics are and how they mix for each document. *Topic Modeling* generalizes this idea of topics over documents. For this case study, we introduce a relatively simple

topic model called *Latent Dirichlet Allocation* (LDA).

In LDA, a *topic* as a concept, which is formulated from the words in the dictionary C . Namely, a topic is an unobserved vector of weights, where the j 'th entry corresponds to word C_j . For example, a topic $\mathbf{s}_1$ which pertains to the concept of a severe accident could attribute high weights to each of the words “injur”, “ambulance”, and “emergency”, whereas a topic $\mathbf{s}_2$ pertaining to a large oil spill could attribute a high weight to “oil”.

Also in LDA, we assume that each *document* pertains to an unobserved mixture of the topics themselves. For instance, a police report about a fatal oil spill may pertain to a topic mixture, in which each of topics $\mathbf{s}_1$ and $\mathbf{s}_2$ is weighted as 40%. Finally in LDA, we assume that the mixture of topics per document yields a probability distribution, from which every *word* in the document is drawn.

Before rigorously formulating all the above-mentioned components of LDA, we note that although LDA might at first seem to be an overly simplistic probabilistic model, it is nevertheless powerful and effective. Indeed, for this case study, LDA yields numeric encodings which greatly improve regression results, as Figure A.18 on page 192 shows. Furthermore, LDA admits completely *unsupervised learning*, as none of the documents is actually labeled with any topics.

A.5.3.1 Formal Definition

For LDA, we need only choose and fix the number of topics K . To reduce dimensionality, K should be chosen as much smaller than the length $|C|$ of the dictionary, so for this case study of incidents, we choose $K = 25$. We denote the latent topics as normalized vectors $\Phi_1, \dots, \Phi_K$, each of length $|C|$.

Next, for each document $d = 1..D$, we define the topic mixture as a linear combination of $\Phi_1, \dots, \Phi_K$, and let latent vector θ_d denote the corresponding linear coefficients.

Finally, we let N_d denote the number of words in document d , and let $w_{d,n}$ denote the n 'th word in d . Interchangeably, $w_{d,n}$ is a number in $1..|C|$, indicating the index of the corresponding word in the dictionary. Word $w_{n,d}$ is then generated in two steps. First, a topic $z_{d,n}$ is chosen per the topic mixture θ_d of the document d . Then, $w_{n,d}$ is drawn per the word proportions in the chosen topic $z_{d,n}$.

The resulting probabilistic model is M_{LDA} , in which the latent parameters to be inferred are $\Phi_1, \dots, \Phi_K$ for the topics, and $\theta_1, \dots, \theta_D$ for the documents. The corresponding PGM appears in Figure A.20 on the next page, and the generative

Figure A.20: PGM for M_{LDA}

Given $\alpha \in \mathbb{R}^K$, $\beta \in \mathbb{R}^{|C|}$, and D documents:

1. For each topic $k = 1, \dots, K$:
 - 1.1. Draw word proportions $\Phi_k \sim \text{Dir}(\Phi_k \mid \beta)$.
2. For each document $d = 1, \dots, D$:
 - 2.1. Draw topic proportions $\theta_d \sim \text{Dir}(\theta_d \mid \alpha)$.
 - 2.2. For each word $n = 1, \dots, N_d$:
 - 2.2.1. Draw topic $z_{d,n} \sim \text{Cat}(z_{d,n} \mid \theta_d)$.
 - 2.2.2. Draw word $w_{d,n} \sim \text{Cat}(w_{d,n} \mid \Phi_{z_{d,n}})$.

Figure A.21: Generative process for M_{LDA}

process is in Figure A.21. A Stan program for M_{LDA} is available in section 17.5 of (Team, 2018).

In Latent Dirichlet Allocation, the priors are Dirichlet distributions, hence the model name. For any document d , $\alpha \in \mathbb{R}^K$ parameterizes the prior on topic mixture θ_d , while for any topic k , $\beta \in \mathbb{R}^{|C|}$ parameterizes the prior on word mixture Φ_k . It follows from the dimension of α that for any document d , θ_d is of relatively low dimension $K \ll |C|$.

This concludes the definition of M_{LDA} . The joint distribution of M_{LDA} has no analytical solution, hence LDA is carried out through approximate Bayesian inference. For example, we could use Variational Inference (VI) (Wainwright and Jordan, 2008), which yields the approximate parameters for the posterior distributions. The posterior distributions of θ_d and Φ_k are also Dirichlet, parameterized on some other vectors from $\mathbb{R}^K$ and $\mathbb{R}^{|C|}$, respectively.

In particular, for any document $d = 1..D$, we can obtain from VI an approximation of parameter γ_d for the posterior Dirichlet distribution of θ_d . γ_d is of low dimension $K \ll |C|$ – much shorter than BoW encoding – and expresses a

posterior belief about topic proportions in d . Hence for regression purposes, we can use γ_d as the desired, short numeric encoding of d .

A.6 Summary

Through several case studies, we have described the Model-Based approach to Machine Learning (MBML), which provides a systematic framework for approaching any modeling problem. The first step in MBML is to formulate our assumptions about the *uncertainty* structure of the problem as a stochastic model, which we express through a Probabilistic Graphical Model (PGM) and a generative process. MBML thus contrasts with traditional Machine Learning approaches, which require us to first select the learning algorithm among a multitude of options.

The following step in MBML is to infer the posterior distribution of unknown parameters in the model, given observed evidence. Subsequently, it is often straightforward to apply model improvements, as well as incorporate new evidence, if any.

In MBML, the posterior distribution is learnt through Bayesian Inference, either exactly or approximately. A popular tool for Bayesian Inference is Stan, and we have provided examples of Stan programs for the various models we built in the case studies. These models apply to a wide range of common modeling problems, including regression, classification, time series, and textual data.

We have limited this chapter to the core concepts of Model-Based Machine Learning, and concentrated on demonstrating the ease and applicability of this approach. However, MBML reaches far beyond this constrained review, and offers a radically different alternative to traditional Machine Learning techniques. It thus comes as no surprise that MBML is currently a very active subject of research and software development.

A.6.1 Further Reading

This chapter is based on notes of course “Model-Based Machine Learning” by co-authors F. Pereira and F. Rodrigues (Pereira and Rodrigues, 2018), which encompasses many additional aspects of MBML. Some parts of this chapter are also adapted from the PhD thesis of F. Rodrigues (Rodrigues, 2016).

We encourage the enthusiastic reader to extend their knowledge into additionally important MBML concepts, as following. Conjugate priors and the exponential

family of distributions play important roles in Bayesian Inference (Bernardo and Smith, 2009, Duda et al., 2012). There exist many methods for approximate Bayesian Inference, such as Markov Chain Monte Carlo (Gilks, 2005), or Variational Inference (Jordan et al., 1999, Wainwright and Jordan, 2008). There are also methods for exact Bayesian Inference, such as Belief Propagation (Pearl, 2014). Probabilistic Graphical Models other than directed Bayesian networks are available, such as unordered graphs and factor graphs (Kschischang et al., 2001). Finally, there is ongoing research into exciting combinations of probabilistic models with Deep Learning (Wang and Yeung, 2016).

Traffic Prediction with Convolutional Long Short-Term Memory

In my first year of PhD studies, I attended MSc course “Deep Learning”, for which I wrote the following project report. The course supervisor was Prof. Ole Winther of the Applied Mathematics and Computer Science Department in DTU, who was impressed with my project and so registered it as a special PhD course for me. I subsequently presented this research output (co-authored with Prof. Winther and my main supervisor, Prof. Francisco C. Pereira) in hEART2018: the 7th Symposium of the European Association for Research in Transportation (Peled et al., 2018).

This work was inspired by Shi et al. (2015), who found that for weather prediction, a relatively new recurrent unit called “Convolutional Long Short-Term Memory” (Conv-LSTM) can take better advantage of spatio-temporal correlations vs. plain Fully-Connected LSTM. We applied a similar idea in the Transportation domain via a dataset of speeds and flows, collected from Android devices in vehicles near Nørre Campus in Copenhagen, Denmark. Starting with a simple Recurrent Neural Network, we gradually enhanced the architecture and eventually obtained that Conv-LSTM again yielded the best predictive performance.

I enjoyed the challenge of applying a recently published method in a very different domain, where it again turned out to be advantageous. Within the theme of my PhD thesis, this work promotes a relatively new Machine Learning method (Conv-LSTM) as a means of improving the accuracy of traffic predictions.

B.1 Introduction

Accurate prediction of traffic conditions enables reliable planning of travel times, early detection of traffic congestion, and effective response by road practitioners. These benefits thus also have positive socio-economic impact.

In this work, we face the challenge of predicting traffic speeds and flows in roads surrounding Nørre Campus – a campus of the University of Copenhagen in Denmark’s capital city. To this end, we design and implement several Neural Network architectures, which outperform common baseline models for traffic

prediction.

We carry out the implementation in Python, mainly using packages `scikit-learn` and `keras`. The code for this work is available in https://github.com/inonpeled/submission_clone_deep_learning_norreccampus.

B.2 Data Description and Preparation

The dataset consists of traffic information for 294 unique places on roads around Nørre Campus in Copenhagen, Denmark, as described in Figure B.1 on the next page. The data was collected by Google from Android devices between 01-Jan-2015 00:00 and 29-Jun-2015 23:59, i.e., the first half of 2015. This time period is partitioned as consecutive, 5 min long intervals, namely 01-Jan-2015 00:00, 01-Jan-2015 00:05, ..., 29-Jun-2015 23:55.

For each place p and time interval t , the dataset contains the mean speed in km/h, and the mean flow decile in $\{0, 1, \dots, 9\}$, as measured in p during t . For example, a flow value of 4 means that the average flow in p during t is greater than at least 40% of the average flows in all other places during t , and smaller than at least 50% of them. The dataset is therefore somewhat noisy: it is aggregated, and some time intervals are missing for various places. Moreover, whereas speed is continuous and pertains locally to only one place, flow decile is discrete and is relative to the global road network.

From here on, we abbreviate "mean flow decile" and "mean speed" simply as "flow" and "speed", respectively. For each place p , we let S_p denote the time series of speeds for p , and let F_p denote the time series of flows for p . Finally, we partition the data into train and test sets: all our models use January to May data for training, while the prediction quality of all our models is tested on June data.

Because traffic in nearby places is often highly correlated, and in the interest of decreasing computation times, we narrow down the data to several selected places, as also described in Figure B.1 on the following page. The selected places are partitioned into two groups, which we study separately: junctions and middle-of-roads (both traffic directions). Because traffic tends to flow more freely in middle-of-roads than in junctions, we may anticipate that prediction models perform better in middle-of-roads than in junctions.

Figure B.1: Mean speed in each of the 294 unique places available in the dataset. The color scale is red for higher values, blue for lower values. Yellow markers indicate selected junctions, and red markers indicate selected middle-of-roads.

B.2.1 Data Cleaning

Before processing the data, we clean it as following. We remove speed outliers above 110 km/h, as the speed limit in the roads around Nørre Campus is much lower. Then, as is common for time series data, we detrend the speeds. That is, for each place p , day $d \in \{\text{Sunday, Monday}, \dots, \text{Saturday}\}$, and time of day $h \in \{00:00, 00:05, \dots, 23:55\}$, we compute the average speed $\mu_{p,d,h}$ and the standard deviation $\sigma_{p,d,h}$ in the train set, and then transform S_p as

$$S'_p = (S_p - \mu_{p,d,h}) / \sigma_{p,d,h}. \quad (\text{B.1})$$

Thus whereas speeds in S_p are non-negative, detrended values in S'_p can be negative. To complete the cleaning, we interpolate missing time intervals in both S'_p and F_p , and round all interpolated flows to the nearest integer in $[0..9]$.

Finally, we wish to convert the clean data into input for the prediction models. To this end, we convert each time series to a matrix of feature vectors, so that the i 'th row vector v_i corresponds to the i 'th time interval t_i . v_i consists of the target variable – i.e., the speed or the flow during t_i – followed by the values for $t_{i-1}, t_{i-2}, \dots, t_{i-12}$, namely the values for the past 60 min.

B.2.2 From Output to Prediction

Our models for speed regression take as input detrended speeds, hence they also output detrended speeds. For measuring speed prediction accuracy, we thus first re-trend the output – namely apply transformation (B.1) in reverse – and remove all interpolated time intervals.

Our models for flow regression treat flow as a continuous variable. Thus for measuring flow prediction accuracy, we first round the output to the nearest integer, and remove all interpolated time intervals. In all our experiments, every such rounded value ended in $\{0, 1, \dots, 9\}$, i.e., the original domain for flow.

In fact, because the categories for flow are ordinal – namely $0 < 1 < \dots < 9$ – we also checked whether flow prediction improves when ordinal regression is used. For Neural Networks, we used the ordinal regression scheme described in (Cheng, 2007), so that: input $k \in \{0, \dots, 9\}$ is provided as a binary vector comprising of $k + 1$ ones followed by $9 - k$ zeros; the output layer comprises of 10 sigmoid neurons; and output $[o_0, \dots, o_9]$ is interpreted as $\max\{i | o_0, \dots, o_i > 0.5\}$. However, the difference in prediction accuracy compared to rounding turned out to be of order $1e-3$, hence we decided to stick with the simpler scheme of rounding.

B.3 Baseline

As baseline for comparison of prediction quality, we build three candidate models: Naive Copy, Historical Average, and Linear Regression. We next describe how each of these models predicts the target variable v_i (speed or flow) at time interval t_i , for any place.

1. *Naive Copy* simply predicts $\tilde{v}_i = v_{i-1}$.
2. *Hisotrical Average* predicts $\tilde{v}_i$ as the average of all $v_{i-1}, v_{i-2}, \dots$.
3. *Linear Regression* (LR) predicts

$$\tilde{v}_i = \beta_0 + \beta_1 v_{i-1} + \dots + \beta_{i-12} v_{i-12}, \quad (\text{B.2})$$

where the β 's are real-valued parameters, which LR learns during training.

The only candidate baseline model which requires training is thus LR. The quality of each candidate model is measured for each place group $G \in \{\text{junctions, middle-of-roads}\}$, as following. For $p \in G$, let $v_1^{(p)}, \dots, v_k^{(p)}$ denote the vectors

in the test set for p , and let $A^{(p)} = a_1^{(p)}, \dots, a_k^{(p)}$ denote the actual values of the target variable, respectively. Furthermore, let $\tilde{V}^{(p)} = \tilde{v}_1^{(p)}, \dots, \tilde{v}_k^{(p)}$ denote the respective predictions of the model for the target variable. We summarize the prediction quality of the model in four measurements:

1. Root Mean Squared Error (lower is better):

$$\text{RMSE} = \frac{1}{|G|} \sum_{p \in G} \left(\sqrt{\frac{1}{k} \sum_{i=1}^k \left(a_i^{(p)} - \tilde{v}_i^{(p)} \right)^2} \right). \quad (\text{B.3})$$

2. Mean Absolute Error (lower is better):

$$\text{MAE} = \frac{1}{|G|} \sum_{p \in G} \left(\frac{1}{k} \sum_{i=1}^k \left| a_i^{(p)} - \tilde{v}_i^{(p)} \right| \right). \quad (\text{B.4})$$

3. Pearson Correlation Coefficient (higher is better):

$$\rho = \frac{1}{|G|} \sum_{p \in G} \left(\frac{\text{Cov} \left(A^{(p)}, \tilde{V}^{(p)} \right)}{\sigma \left(A^{(p)} \right) \sigma \left(\tilde{V}^{(p)} \right)} \right), \quad (\text{B.5})$$

where Cov denotes covariance, and σ denotes standard deviation.

4. Coefficient of Determination (higher is better):

$$R^2 = \frac{1}{|G|} \sum_{p \in G} \left(1 - \frac{\sum_{i=1}^k \left(a_i^{(p)} - \tilde{v}_i^{(p)} \right)^2}{\sum_{i=1}^k \left(a_i^{(p)} - \bar{A}^{(p)} \right)^2} \right), \quad (\text{B.6})$$

where $\bar{A}^{(p)}$ is the mean of $A^{(p)}$.

Table B.1 on the next page and Table B.2 on the facing page summarize the prediction quality of all three candidate baseline model, while Figure B.2 on the next page visually compares their RMSE. All candidate baseline models perform worse in junctions than in middle-of-roads, as we expected. Linear Regression performs consistently the best, while Naive Copy is nearly as good as LR, which suggests that traffic flows and speeds tend to repeat themselves over consecutive, short time intervals. Finally, Historical Average turns out to be a rather unreliable prediction model for our traffic data.

We thus keep only Linear Regression as the baseline for later comparison with the Neural Networks. Next, we check whether the prediction quality of LR

	Place Group	HistAvg	LR	Copy
RMSE	Middle	7.913	4.758	5.241
	Junction	10.267	6.719	7.501
MAE	Middle	5.370	2.477	3.018
	Junction	7.348	3.893	4.359
ρ	Middle	0.332	0.829	0.808
	Junction	0.173	0.768	0.749
R^2	Middle	0.099	0.674	0.606
	Junction	0.017	0.579	0.476

Table B.1: Performance of candidate baseline models for speed only. The best value in each row is highlighted in bold.

	Place Group	HistAvg	LR	Copy
RMSE	Middle	1.741	0.568	0.583
	Junction	1.941	0.772	0.802
MAE	Middle	1.393	0.274	0.282
	Junction	1.572	0.468	0.486
ρ	Middle	0.024	0.943	0.941
	Junction	0.0153	0.913	0.908
R^2	Middle	negative	0.888	0.882
	Junction	negative	0.831	0.816

Table B.2: Performance of candidate baseline models for flow only. The best value in each row is highlighted in bold.

improves if it is trained on speeds and flows together. It makes sense to try so, because speeds and flows are usually correlated, e.g., high congestion leads to low flow and low speed. However, Table B.3 on the following page shows that LR performs essentially the same whether or not it learns speeds and flows are together.

Figure B.2: RMSE of the three baseline candidate models. A star marks the best value in each case.

	Place Group	Speed	Flow
RMSE	Middle	4.757 (-0.001)	0.546 (-0.022)
	Junction	6.714 (-0.005)	0.735 (-0.043)
MAE	Middle	2.477 (0)	0.325 (+0.051)
	Junction	3.893 (0)	0.519 (+0.051)
ρ	Middle	0.829 (0)	0.947 (+0.003)
	Junction	0.769 (+0.001)	0.921 (+0.008)
R^2	Middle	0.674 (0)	0.896 (+0.008)
	Junction	0.579 (0)	0.846 (+0.015)

Table B.3: Performance of Linear Regression when speeds and flows are provided together. In parenthesis is the difference from the performance of LR when speeds and flows are provided separately.

Figure B.3: Neural Network architectures in our experiments.

B.4 Deep Neural Networks

In this section, we’ll study Deep Neural Networks (NN) for prediction of speeds and flows. We’re interested in two research questions: 1) Are NN also sensitive to the differences between middle-of-roads and junctions? 2) how much better can NN outperform the baseline?

Our basic architecture of choice is Recurrent Neural Network (RNN), which is a natural choice for sequential data (Karpathy, 2015). However, some online resources (e.g., (Brownlee, 2017)) suggest that RNN may not be easily applicable to time series data. Nevertheless, we will start from a simple RNN architecture and gradually enhance it, in an attempt to improve the prediction quality. Figure B.3 illustrates some of the architectures we experiment with, while Tables B.4 and B.5 in Section B.4.9 summarize the results of all experiments, as we detail next.

B.4.1 Selection of Hyper-Parameters

In order to select the best hyper-parameters for use in all NN, we perform exhaustive search over all the following options: LSTM state size in $\{10, 20, 30\}$; number of past lags in $\{3, 6, 12\}$; mini-batch size in $\{256, 512\}$; number of epochs in $\{50, 100, 150\}$. For each combination of these options, we train a simple RNN, as depicted in Figure B.3a on the facing page, and measure its performance on speeds in 6 places: 3 middle of roads and 3 junctions, all chosen arbitrarily.

Figure B.4 on the next page summarizes the results of the exhaustive search. We conclude that LSTM state size has the most consistent effect on performance, so that more cells yield better prediction quality. Also, the smaller mini-batch size yields better results, and better performance is gained by learning less past lags. Intuitively, the speed in the immediate future is more affected by speeds in the immediate past, rather than speeds in the distant past.

Thus from here on, all our NNs will be built with the following hyper-parameters, unless otherwise stated: LSTM state size = 30; past lags = 3; mini-batch size = 256; epochs = 100. We could otherwise choose 150 epochs for the same performance, but having more epochs increases the risk of over-fitting to train data.

B.4.2 Neural Networks with FC-LSTM

We begin by constructing RNN's where the recurrent units are "classic", Fully Connected Long Short-Term Memory (FC-LSTM) units. As mentioned before, Table B.4 on page 209 and Table B.5 on page 210 summarize the performance of these models, and compares them to the baseline LR. Furthermore, Figure B.5a on page 207 and Figure B.5b on page 207 compare the RMSE of all models visually. These are the tables and figures we refer to in the remainder of this section.

B.4.3 LSTM with Separated Input

Our first and simplest architecture is illustrated in Figure B.3a on the facing page. We name this architecture *LSTM Separated*, because it is applied separately to speeds and to flows. We run this NN once for each selected place, and then aggregate the results as described in the formulae for prediction quality measurements. The tables and figures show that LSTM Separated is somewhat

<i>LstmStateSize</i>	<i>NumLagsBack</i>	<i>MiniBatchSize</i>	<i>Epochs</i>	<i>corr</i>	<i>mae</i>	<i>R^2</i>	<i>rmse</i>
30	3	256	150	0.81	2.87	0.65	5.21
30	3	256	100	0.81	2.87	0.65	5.21
20	3	256	150	0.81	2.87	0.65	5.22
30	6	256	150	0.81	2.87	0.65	5.22
30	12	256	150	0.81	2.88	0.65	5.22
30	12	256	100	0.81	2.90	0.65	5.22
20	6	256	150	0.81	2.88	0.65	5.22
20	3	256	100	0.81	2.88	0.65	5.22
10	3	256	150	0.81	2.88	0.65	5.23
30	6	512	150	0.81	2.90	0.65	5.23
30	3	512	150	0.81	2.87	0.65	5.23
30	6	256	100	0.81	2.90	0.65	5.23
20	6	256	100	0.81	2.91	0.65	5.23
10	3	512	150	0.81	2.89	0.65	5.23
20	3	512	150	0.81	2.89	0.65	5.24
10	3	256	100	0.81	2.89	0.65	5.24
20	3	512	100	0.81	2.90	0.65	5.24
30	3	512	100	0.81	2.90	0.65	5.24
20	6	512	150	0.81	2.92	0.65	5.24
20	12	256	150	0.81	2.89	0.65	5.24
30	12	512	150	0.81	2.93	0.65	5.25
10	6	256	150	0.81	2.92	0.65	5.25
20	12	256	100	0.81	2.93	0.65	5.25
30	3	256	50	0.81	2.92	0.65	5.25
10	3	512	100	0.81	2.92	0.65	5.26
10	6	256	100	0.81	2.93	0.65	5.26
20	12	512	150	0.81	2.94	0.65	5.26
20	3	256	50	0.81	2.94	0.65	5.26
30	6	512	100	0.81	2.95	0.65	5.27
30	12	512	100	0.81	2.99	0.65	5.27
30	6	256	50	0.81	2.96	0.65	5.27
10	3	256	50	0.81	2.93	0.65	5.27
30	3	512	50	0.81	2.94	0.65	5.27
20	6	512	100	0.81	2.95	0.65	5.27
10	12	256	150	0.80	2.96	0.64	5.29
20	3	512	50	0.81	2.96	0.64	5.29
10	6	512	150	0.81	2.96	0.64	5.29
20	12	512	100	0.80	3.02	0.64	5.29
10	6	512	100	0.80	2.99	0.64	5.29
30	12	256	50	0.80	3.02	0.64	5.30
10	6	256	50	0.80	3.03	0.64	5.30
10	12	256	100	0.80	2.99	0.64	5.30
20	6	256	50	0.80	2.99	0.64	5.31
10	3	512	50	0.80	2.96	0.64	5.31
10	12	512	150	0.80	3.00	0.64	5.31
30	6	512	50	0.80	3.05	0.64	5.33
10	6	512	50	0.80	3.08	0.64	5.33
20	12	256	50	0.80	3.05	0.64	5.34
10	12	512	100	0.80	3.11	0.63	5.37
20	6	512	50	0.80	3.08	0.63	5.38
30	12	512	50	0.79	3.18	0.63	5.39
10	12	256	50	0.79	3.19	0.63	5.42
20	12	512	50	0.79	3.24	0.62	5.47
10	12	512	50	0.77	3.39	0.60	5.61

Figure B.4: Exhaustive search for best hyper-parameters. The color scale is blue for better values, red for worse values.

better at predicting speeds, but worse at predicting flows, compared to the baseline LR.

B.4.4 LSTM with Combined Input

We thus advance to the next architecture, illustrated in Figure B.3b on page 204. We name this architecture *LSTM Combined*, because it is applied to speeds and flows together. As before, we run this network on each selected place separately before aggregating the results for each place group. The tables and figures show that LSTM Combined is better than all previous models at predicting both speeds and flows. Hence unlike LR, LSTM can take advantage of processing speeds or flows together rather than separately.

Figure B.5: RMSE for all models. A star marks the best value in each case.

B.4.5 Mixture of Experts

Because combining speeds and flows improved the prediction quality, we next check whether it is better to first specialize one LSTM for each of the two types of data, and only then combine the learned representations. Figure B.3c on page 204 illustrates this architecture, which we name *LSTM Mixture*, because it employs a mixture of expert LSTM's. We run this network twice for each place: once with LSTM state size 15 – so that the total state size is 30 as before – and once with state size 30 – so that the network has twice as much memory. The tables show that even with twice the memory, the mixture of experts performs worse than LSTM Combined, which is therefore still the best architecture so far. Interestingly also, LSTM Mixture performs virtually the same regardless of the size of LSTM state.

B.4.6 Dropout and Regularization

As LSTM Combined is still the best performing architecture so far, we next try to improve it by introducing dropout and regularization. These techniques may reduce overfitting on the train set, and so yield better performance on the test set. We try out all combinations of the following parameters: dropout rate in $\{0, 0.25, 0.5, 0.75\}$, applied to the LSTM unit; kernel regularizer in $\{\text{None}, l_2(1e-2), l_2(1e-4), l_2(1e-6)\}$, applied to the dense layer. The baseline LSTM Combined is obtained for dropout rate zero and no regularization. We obtain that performance only worsens as more dropout and more regularization are applied, and so conclude not to use these techniques.

B.4.7 LSTM with Entire Place Group Input

For the last RNN architecture that we try with FC-LSTM, we feed an entire place group $G \in \{\text{middle-of-roads}, \text{junctions}\}$ at once, as in Figure B.3d on page 204. This way, perhaps the NN could take advantage of an overview of the road network. We name this architecture *LSTM Grouped*, and run it first with the same hyper-parameters as before. The results appear in the tables under LSTM Grouped₁, and show worse performance than LSTM Combined, which runs on one place at a time.

Thereafter, we increase the number of epochs to 150 and the state size to $30|G|$, and run again. The results for these larger hyper-parameters appear in the tables under LSTM Grouped₂. The performance is similar to LSTM Combined, with a slight improvement in junctions.

B.4.8 Convolutional LSTM

So far, our models have been oblivious to the structure of the underlying road network in the data. This structure should account for spatio-temporal correlations between different places, e.g., traffic congestion in a junction quickly spills over to the roads surrounding the junction. We now wish to build a model which takes advantage of such spatio-temporal correlations.

Our inspiration comes from (Shi et al., 2015), which shows that Convolutional LSTM outperforms FC-LSTM in short-term prediction of weather over a 2-dimensional grid. In essence, a Convolutional LSTM (Conv-LSTM) unit internally applies convolution instead of matrix product, and so can learn patterns in local neighborhoods. In contrast, Fully-Connected LSTM (FC-LSTM) is equipped to learn global patterns.

Based on LSTM Grouped, we thus create a new architecture by replacing FC-LSTM with Conv-LSTM, as illustrated in Figure B.3e on page 204. Next, we note that placing a 2D grid on the map of places could be problematic in several aspects. First, the selected places would occupy only a few cells in the grid, which would thus be highly sparse. Second, neighboring cells in the grid would contain places on different traffic directions for the same road, which could hinder learning.

Thus instead of feeding our newest architecture proper 2D input, we first reduce the input to 1D, hence we name this architecture *LSTM Conv1D*. That is, we order all places linearly, and run the NN separately for each ordering, which are: random, by latitude (i.e., horizontally), and by longitude (i.e., ver-

tically). The results are summarized in the tables under LSTM Conv1D_{RND}, LSTM Conv1D_{LAT}, and LSTM Conv1D_{LNG}, respectively. In all experiments, the convolutional kernel size is 5×1 .

The results show that for our data too, Conv-LSTM outperforms FC-LSTM in predicting speeds and flows, regardless of which linear ordering we try. This suggests that the improvement is brought about by the use of convolution instead of matrix product in the LSTM. Furthermore, ordering by latitude or by longitude yields better performance than ordering randomly, which suggests that our latest architecture indeed takes advantage of structural properties. The best improvement is gained for ordering by longitude; Figure B.1 on page 200 indeed shows that the selected places are more widely scattered vertically than horizontally.

B.4.9 Summary of Results

Tables B.4 and B.5 hereby summarize the results of all experiments, as detailed in previous Sections.

Model	Place Group	RMSE	MAE	ρ	R^2
Baseline LR	Middle	4.757	2.477	0.829	0.674
	Junction	6.714	3.893	0.769	0.579
LSTM Separated	Middle	4.521 (-0.236)	2.383 (-0.094)	0.846 (+0.017)	0.614 (+0.060)
	Junction	6.436 (-0.278)	3.735 (-0.158)	0.787 (+0.018)	0.614 (+0.035)
LSTM Combined	Middle	4.263 (-0.494)	2.313 (-0.164)	0.862 (+0.033)	0.740 (+0.066)
	Junction	6.212 (-0.502)	3.634 (-0.259)	0.802 (+0.033)	0.642 (+0.063)
LSTM Mixture ₁₅	Middle	4.528 (-0.229)	2.389 (-0.088)	0.845 (+0.016)	0.706 (+0.032)
	Junction	6.425 (-0.289)	3.72 (-0.173)	0.788 (+0.019)	0.615 (+0.036)
LSTM Mixture ₃₀	Middle	4.517 (-0.240)	2.378 (-0.099)	0.846 (+0.017)	0.708 (+0.034)
	Junction	6.422 (-0.292)	3.713 (-0.180)	0.789 (+0.020)	0.616 (+0.037)
LSTM Grouped ₁	Middle	5.729 (+0.972)	3.511 (+1.034)	0.729 (-0.100)	0.524 (-0.150)
	Junction	6.537 (-0.177)	3.956 (+0.063)	0.778 (+0.009)	0.601 (+0.022)
LSTM Grouped ₂	Middle	4.543 (-0.214)	2.357 (-0.120)	0.847 (+0.018)	0.704 (+0.030)
	Junction	6.376 (-0.338)	3.782 (-0.111)	0.792 (+0.023)	0.622 (+0.043)
LSTM Conv1D _{RND}	Middle	4.22 (-0.537)	2.314 (-0.163)	0.867(+0.038)	0.746 (+0.072)
	Junction	6.163 (-0.551)	3.6 (-0.293)	0.807(+0.038)	0.648 (+0.069)
LSTM Conv1D _{LAT}	Middle	4.209 (-0.548)	2.294 (-0.183)	0.866 (+0.037)	0.747 (+0.073)
	Junction	6.161 (-0.553)	3.593(-0.300)	0.807(+0.038)	0.648 (+0.069)
LSTM Conv1D _{LNG}	Middle	4.184(-0.573)	2.293(-0.184)	0.867(+0.038)	0.750(+0.076)
	Junction	6.141(-0.573)	3.607 (-0.286)	0.807(+0.038)	0.651(+0.072)

Table B.4: Speed prediction quality for all models. Improvement over baseline is in parenthesis. Best values are in bold.

Model	Place Group	RMSE	MAE	ρ	R^2
Baseline LR	Middle	0.546	0.325	0.947	0.896
	Junction	0.735	0.519	0.921	0.846
LSTM Separated	Middle	0.557 (+0.011)	0.270 (-0.055)	0.945 (-0.002)	0.893 (-0.003)
	Junction	0.753 (+0.018)	0.463 (-0.056)	0.917 (-0.004)	0.840 (-0.006)
LSTM Combined	Middle	0.539 (-0.007)	0.327 (+0.002)	0.949 (+0.002)	0.900 (+0.004)
	Junction	0.718 (-0.017)	0.518 (-0.001)	0.925 (+0.004)	0.855 (+0.009)
LSTM Mixture ₁₅	Middle	0.539 (-0.007)	0.327 (+0.002)	0.948 (+0.001)	0.899 (+0.003)
	Junction	0.718 (-0.017)	0.516 (-0.003)	0.924 (+0.003)	0.855 (+0.009)
LSTM Mixture ₃₀	Middle	0.539 (-0.007)	0.326 (+0.001)	0.949 (+0.002)	0.9 (+0.004)
	Junction	0.717 (-0.018)	0.514 (-0.005)	0.925 (+0.004)	0.855 (+0.009)
LSTM Grouped ₁	Middle	0.66 (+0.114)	0.461 (+0.136)	0.922 (-0.025)	0.848 (-0.048)
	Junction	0.726 (-0.009)	0.542 (+0.023)	0.924 (+0.003)	0.851 (+0.005)
LSTM Grouped ₂	Middle	0.537 (-0.009)	0.364 (+0.039)	0.95 (+0.003)	0.9 (+0.004)
	Junction	0.718 (-0.017)	0.541 (+0.022)	0.926 (+0.005)	0.854 (+0.008)
LSTM Conv1D _{RND}	Middle	0.517 (-0.029)	0.334 (+0.009)	0.953 (+0.006)	0.907 (+0.011)
	Junction	0.693 (-0.042)	0.51 (-0.009)	0.930 (+0.009)	0.864 (+0.018)
LSTM Conv1D _{LAT}	Middle	0.517 (-0.029)	0.332 (+0.007)	0.953 (+0.006)	0.907 (+0.011)
	Junction	0.693 (-0.042)	0.509 (-0.010)	0.930 (+0.009)	0.864 (+0.018)
LSTM Conv1D _{LNG}	Middle	0.515 (-0.031)	0.327 (+0.002)	0.953 (+0.006)	0.908 (+0.012)
	Junction	0.692 (-0.043)	0.504 (-0.015)	0.930 (+0.009)	0.865 (+0.019)

Table B.5: Flow prediction quality for all models. Improvement over baseline is in parenthesis. Best values are in bold.

B.5 Conclusions and Future Work

Starting from a simple RNN based on FC-LSTM, we gradually enhanced the architecture to predict speeds and flows better than a baseline LR model. When we reached a point where performance improvements became marginal, we turned to use Conv-LSTM, which noticeably outperformed FC-LSTM. In comparison with the predictions of the baseline LR, our best RNN achieves 12% better speed RMSE in middle-of-roads, 8.5% better speed RMSE in junctions, 5.7% better flow RMSE in middle-of-roads, and 5.9% better flow RMSE in junctions.

We saw that similarly to LR, all our RNN models yielded more accurate predictions in middle-of-roads than in junction. We also saw that unlike LR, RNN could take advantage of combining speeds and flows at input level.

For future work, Graph Convolutional LSTM (Seo et al., 2016) appears promising for prediction of traffic in road networks. Our data is naturally structured as a graph, such that places are vertices, and edges exist between places which reside on the same road segment. Moreover, edges may be weighted by, e.g., the length of the corresponding road segment. We can also try to improve prediction accuracy by stacking recurrent units – a common technique for adding levels of abstraction in RNN.