

Real-Time Short-Term Intersection Turning Movement Flows Forecasting using Deep Learning Models for Advanced Traffic Management and Information Systems

by

Ce Zhang

A thesis
presented to University of Waterloo
in fulfillment of the
thesis requirements for the degree of
Doctor of Philosophy
in
Civil Engineering

Waterloo, Ontario, Canada, 2025

© Ce ZHANG 2025

Examining committee membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner	Xuegang (Jeff) BAN Professor
Supervisor(s)	Liping FU Professor
Internal Member	Chu Min Yeum Professor
Internal Member	Pejoohan Tavassoti-Kheiry Professor
Internal-external Member	Mu ZHU Professor

Author's declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of contributions

Ce Zhang was the sole author of Chapters 1, 2, and 7, which were written under the supervision of Dr. Liping Fu and were not intended for publication.

This thesis consists in part of four manuscripts written for publication. Exceptions to sole authorship of material are as follows:

Research presented in Chapter 3

This research was conducted at the University of Waterloo by Ce Zhang under the supervision of Dr. Liping Fu. Ce Zhang designed the study with input from Dr. Guangyuan Pan and Dr. Liping Fu. Ce Zhang was responsible for data processing, while analysis and interpretation of results were conducted by Ce Zhang, Dr. Guangyuan Pan, and Dr. Liping Fu. The manuscript was drafted by Ce Zhang, with contributions from Dr. Guangyuan Pan, Dr. Liping Fu, and Dr. Chris Bachmann. All authors reviewed the results and approved the final version of the manuscript.

Zhang, C., Pan, G., & Fu, L. (2024). Real-Time Intersection Turning Movement Flow Forecasting Using a Parallel Bidirectional Long Short-Term Memory Neural Network Model. *Transportation Research Record*, 2678(2), 167-183. <https://doi.org/10.1177/03611981231172958>

Research presented in Chapter 4

This research was conducted at the University of Waterloo by Ce Zhang under the supervision of Dr. Liping Fu. Ce Zhang designed the study with input from Dr. Matthew I. Muresan, Dr. Guangyuan Pan, and Dr. Liping Fu. Ce Zhang and Dr. Matthew I. Muresan were responsible for data processing, while analysis and interpretation of results were conducted by Ce Zhang, Dr. Guangyuan Pan, and Dr. Liping Fu. The manuscript was drafted by Ce Zhang, with contributions from Dr. Guangyuan Pan, Dr. Matthew I. Muresan, and Dr. Liping Fu. All authors reviewed the results and approved the final version of the manuscript.

Ce Zhang, G. P., Matthew I. Muresan, Liping Fu (2024). Uncertainty Quantification of Intersection-level Turning Movement Flows Forecasting by Deep Learning Method. 103rd Transportation Research Board Annual Meeting, (Accepted)

Research presented in Chapter 5

This research was conducted at the University of Waterloo by Ce Zhang under the supervision of Dr. Liping Fu. Ce Zhang designed the study with input from Dr. Matthew I. Muresan, Dr. Guangyuan Pan, and Dr. Liping Fu. Ce Zhang and Dr. Matthew I. Muresan were responsible for data processing, while analysis and interpretation of results were conducted by Ce Zhang, Dr. Guangyuan Pan, and Dr. Liping Fu. The manuscript was drafted by Ce Zhang, with contributions from Dr. Guangyuan Pan, Dr. Matthew I. Muresan, and Dr. Liping Fu. All authors reviewed the results and approved the final version of the manuscript.

Zhang, C., Pan, G., Muresan, M. I., Lu, Z., & Fu, L. (2024, September). Temporal Fusion Transformer for Real-Time Intersection Turning Movement Flow Forecasting Incorporating Exogenous Factors. In 2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC) (pp. 1231-1238). IEEE.

Research presented in Chapter 6

This research was conducted at the University of Waterloo by Ce Zhang under the supervision of Dr. Liping Fu. Ce Zhang designed the study with input from Yuzhe You, Dr. Matthew I. Muresan, Dr. Guangyuan Pan, and Dr. Liping Fu. Ce Zhang and Dr. Matthew I. Muresan were responsible for data processing, while analysis and interpretation of results were conducted by Ce Zhang, Yuzhe You, Dr. Guangyuan Pan, and Dr. Liping Fu. The manuscript was drafted by Ce Zhang, with contributions from Yuzhe You, Dr. Guangyuan Pan, Dr. Matthew I. Muresan, and Dr. Liping Fu. All authors reviewed the results and approved the final version of the manuscript.

Ce Zhang, Y. Y., Guangyuan Pan, Matthew I. Muresan, Liping Fu. Real-Time Traffic Turning Flow Forecasting Under Few-Shot and Zero-Shot Scenarios by Decoder-only Model. *Transportmetrica A: Transport Science*, (Under Review)

As the lead author of these four chapters, I was responsible for conceptualizing the study design, carrying out data collection and analysis, and drafting and submitting the manuscripts. My co-authors provided guidance throughout each step of the research and gave feedback on draft manuscripts.

Abstract

Traffic congestion remains a persistent challenge in urban transportation systems, causing excessive travel delays, increased fuel consumption, and severe environmental pollution. To address these issues, Advanced Traffic Management and Information Systems (ATMIS) have been developed, integrating real-time traffic monitoring, adaptive control strategies, and data-driven decision-making to enhance overall traffic efficiency. A crucial component of ATMIS is the real-time forecasting of intersection Turning Movement Flows (TMFs), which provides essential data for optimizing signal timings, improving vehicle routing, and implementing proactive congestion mitigation strategies. By leveraging accurate TMFs predictions, transportation agencies can dynamically adjust traffic signals, enhance intersection operations, and reduce delays, ultimately improving urban mobility and minimizing environmental impacts.

While numerous traffic forecasting models exist, they face significant limitations in capturing the complex spatial and temporal patterns inherent in intersection-level TMFs, as they primarily rely on historical traffic data without adequately modeling these dependencies. Moreover, most existing approaches fail to incorporate exogenous factors, such as weather conditions, road characteristics, and other time-dependent variables, which significantly influence traffic flow but are often ignored. These shortcomings lead to poor generalization performance when applied to hold-out intersections (few-shot) and unseen regions (zero-shot), making them less effective in real-world dynamic traffic environments.

To overcome these challenges, this study systematically develops and evaluates a deep learning-based TMFs forecasting framework designed for improved generalization and interpretability. First, we employ a Parallel Bidirectional LSTM (PB-LSTM) with multilayer perceptron (MLP) to capture both long-term seasonality and spatial dependencies, thereby enhancing the model's transferability across different locations, improving performance across hold-out intersections. Second, we integrate an encoder-decoder architecture using Deep Autoregressive (DeepAR) model, which enables probabilistic forecasting and quantifies uncertainty, ensuring robust predictions under varying traffic conditions. Third, we leverage the Temporal Fusion

Transformer (TFT) to assess the relative importance of external covariates, such as weather conditions and road characteristics, improving interpretability and model reliability by identifying speed zone, road category, hour of the day, and temperature as key influential factors. Finally, we explore the potential of TimesFM, a decoder-only model, to enhance zero-shot learning capabilities, demonstrating strong performance in previously unseen intersections and new city datasets, particularly when enhanced with EMD and RF.

To evaluate model performance, we conduct a series of experiments, including hold-out intersection tests, cross-city generalization assessments, and evaluations under extreme weather conditions, to assess robustness and adaptability. Experimental results highlight the effectiveness of integrating exogenous factors and hybrid modeling approaches in improving real-time TMFs forecasting accuracy, generalizability, and robustness under dynamic conditions. These insights provide valuable contributions to the development of scalable and interpretable deep learning models for intersection-level traffic flow prediction, supporting more adaptive and data-efficient traffic management strategies.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Professor Liping Fu, for his invaluable guidance, encouragement, and support throughout my Ph.D. journey. His expertise and insightful advice have been instrumental in shaping this research and helping me grow as a researcher.

I would also like to extend my deepest appreciation to Dr. Matthew I. Muresan and Dr. Guangyuan Pan for their collaboration and valuable contributions to this work. Their expertise and constructive feedback have significantly enriched the quality of this thesis.

I am grateful to the Department of Civil & Environmental Engineering at the University of Waterloo for providing an excellent research environment and access to valuable resources. I would also like to thank my colleagues and fellow researchers in the Innovative Transportation System Lab (ITSS Lab) for their stimulating discussions, technical support, and encouragement.

Special thanks to Miovision Technologies, City of Milton, and Region of Waterloo for providing access to real-world traffic data, which was essential for the successful completion of this research.

I also acknowledge the financial support from the Natural Sciences and Engineering Research Council of Canada (NSERC), ORF-RE (Ontario Research Fund – Research Excellence) project “Intelligent Systems for Sustainable Urban Mobility (ISSUM), and the University of Waterloo, which made this research possible.

Finally, I am deeply thankful to my family and friends for their unwavering love, patience, and support throughout this journey. Their encouragement has been my source of strength, and I would not have accomplished this without them.

Table of contents

Examining committee membership	ii
Author's declaration	iii
Statement of contributions	iv
Abstract	vii
Acknowledgements	ix
List of Figures	xiv
List of Tables	xvii
Chapter 1: Introduction	1
1.1 Background	1
1.2 Research problem and objectives	2
1.3 Thesis structure	5
Chapter 2: Literature Review	7
2.1 Traffic volume forecasting models	7
2.1.1 Statistical forecasting models	7
2.1.2 Traditional machine learning forecasting models	8
2.1.3 Deep learning forecasting models	11
2.2 Traffic Turning Movement Flows (TMFs) forecasting	15
2.3 Summary	17
Chapter 3: Real-Time Intersection Turning Movement Flow (TMFs) Forecasting using A Parallel Bidirectional Long Short-Term Memory (PB-LSTM) Neural Network Model	19
3.1 Literature review	20
3.2 Methodology	22
3.2.1 Proposed model	22

3.2.2	Case study data description	25
3.3	Computational experiments and results	27
3.3.1.	Preliminary performance evaluation	28
3.3.2.	Overall performance.....	30
3.3.3.	Effect of input data range	34
3.3.4.	Effect of the forecasting horizon	35
3.3.5.	Benefits of location clustering and model transferability	36
3.4	Conclusions.....	40
Chapter 4: A Deep Learning Model for Intersection-Level Turning Movement Flows Forecasting with Uncertainty Quantification.....		42
4.1	Literature review	43
4.2	Problem description and DeepAR model.....	44
4.3	Computational experiments and results	48
4.3.1	Case study data description.....	49
4.3.2	Performance metrics.....	50
4.3.3	Model performance	52
4.3.4	Comparison with Transformer and ARIMA for multistep forecasting.....	54
4.3.5	Effect of time varying covariates	56
4.3.6	Effect of cross-sectional features	57
4.3.7	Transferability analysis with hold-out intersections	59
4.4	Conclusions.....	61
Chapter 5 Temporal Fusion Transformer for Real-Time Intersection Turning Movement Flow Forecasting Incorporating Exogenous Factors		64
5.1	Literature review	65
5.2	Research design and methodology.....	66
5.2.1	Forecasting framework.....	66

5.2.2	Temporal Fusion Transformer architecture.....	68
5.3	Experimental results.....	73
5.3.1	Study area and data description.....	73
5.3.2	Experimental set-up	76
5.3.3	Result analysis and comparison	77
5.4	Conclusions.....	82
Chapter 6: Real-Time Traffic Turning Flow Forecasting under Few-Shot and Zero-Shot Scenarios by Decoder-Only Model		83
6.1	Background and review of literature.....	84
6.1.1	Encoder-decoder models	84
6.1.2	Decoder only models.....	85
6.1.3	Application of encoder-decoder and decoder-only models in traffic flow forecasting.....	86
6.2	Methodology	89
6.2.1	TimesFM.....	91
6.2.2	Encoder-decoder models for comparison.....	93
6.3	Experimental results.....	94
6.3.1	Case study data description	94
6.3.2	Performance measures.....	97
6.3.3	Model performance for different datasets	99
6.3.4	Multistep forecasting comparison	105
6.4	Conclusions.....	107
Chapter 7: Conclusions		109
7.1	Summary of major studies and findings	109
7.2	Summary of major contributions	110
7.3	Recommendations for future research	112

References.....	113
Appendix.....	126

List of Figures

Figure 1.1: The TMFs forecasting problem	4
Figure 3.1: The structure of the basic LSTM cell (Sharma et al., 2021).....	23
Figure 3.2: Stacked BiLSTM structure	24
Figure 3.3: Model Summary	25
Figure 3.4: Case Study Intersections (The brown dots are the intersections being monitored, the blue dots are the intersections selected for analysis, and the intersections in the red circle are additional test intersections)	26
Figure 3.5: Daily TMCs over the analysis period.....	27
Figure 3.6: A comparison of performance of some well-known time-series forecasting models.....	29
Figure 3.7: Comparison of Turning Movements Forecasting Errors by LSTM, SBiLSTM and SBiLSTM with MLP	30
Figure 3.8: Performance of the PB-LSTM model for forecasting turning movement flows: predicted vs. observed daily volume (From 2020 Jan 01 to 2020 Dec 31, 30 mins input data range and 5 mins forecasting horizon)	31
Figure 3.9: Performance of the PB-LSTM Model for forecasting turning movement flows: Predicted vs. Observed 5-min volume (weekdays and weekends, 30 mins input data range and 5 mins forecasting horizon)	32
Figure 3.10: The average RMSE of the PB-LSTM Model and SBiLSTM model for forecasting turning movement flows in eastbound (30 mins input data range and 5 mins forecasting horizon)	33
Figure 3.11: The weighted average performance comparison of 2 models under different observation range when predicting the next 5 mins flow	35
Figure 3.12: The weighted average performance comparison of 3 models for different prediction horizons when inputting 30 mins historical TMFS.	36
Figure 3.13: Classification results (scenario 1: left, scenario 2: middle, scenario 3: right)	37
Figure 3.14: Comparison of PB-LSTM with Geometry classification label, PB-LSTM with approach classification label, PB-LSTM with ID numbers, and typical LSTM models when applied in 20 intersections	38
Figure 3.15: Comparison of PB-LSTM with Geometry classification label, PB-LSTM with approach classification label, PB-LSTM with ID numbers, and typical LSTM models when applied in Ontario Street South and Childs Drive.....	39

Figure 3.16: Comparison of PB-LSTM with Geometry classification label, PB-LSTM with Approach classification label, PB-LSTM with ID numbers, and normal LSTM models when applied to Main Street and Leisure Centre Driveway	40
Figure 4.1: Traffic flow from different directions and turning movements.....	45
Figure 4.2: The TMFs forecasting problem	46
Figure 4.3: DeepAR model architecture(Salinas et al., 2020)	48
Figure 4.4: The study area (left: location of City of Milton; right: the locations of collected intersections)	50
Figure 4.5: Sample of observed and predicted TMFs (vehs/ interval).....	54
Figure 4.6: Forecasting performance comparison between DeepAR, Transformer, and ARIMA with different forecasting horizons	55
Figure 4.7: Forecasting performance for DeepAR model with covariates and without covariates	56
Figure 4.8: Forecasting performance for DeepAR trained by multiple movements and by single movement	58
Figure 4.9: Forecasting performance for DeepAR model on hold-out intersections	60
Figure 5.1: Framework of utilizing the TFT for TMFs forecasting.....	67
Figure 5.2: The structure of Temporal Fusion Transformer(Lim et al., 2021).	68
Figure 5.3: Intersection's location(a) and turning movements(b).....	74
Figure 5.4: Forecasting results of TFT on May 2nd, 2023, at intersection Fairway at Wilson.	78
Figure 5.5: Importance of static covariates for each movement	79
Figure 5.6: Importance of time-dependent covariates for learning (a) and forecasting (b)	80
Figure 5.7: TFT forecasting performance based on exogenous factors.	81
Figure 6.8: Encoder-Decoder and decoder-only time series forecasting models.....	84
Figure 6.9: Traffic flow from different directions and turning movements.....	89
Figure 6.10: Research Methodology	90
Figure 6.11: EMD example on TMFs dataset.....	92
Figure 6.12: Three alternative TimesFM models.....	93
Figure 6.13: Case study sites	95

Figure 6.14: Intrumented intersections at the two locations	96
Figure 6.15: Sample of observed and predicted TMFs.	102
Figure 6.16: TimesFM Forecascting EMD decomposition examples on Dec 2nd, 2023, at intersection Fairway at Wilson.	104
Figure 6.17: RMSE, RMSPE, MAE and VAPE for different forecasting horizons.	105

List of Tables

Table 3.1: The features included in the raw data	26
Table 3.2: The RMSE of PB-LSTM for Forecasting Turning Movement Flows of 20 Intersections	34
Table 3.3: K-means classification scenarios	37
Table 3.4: Model comparison in Intersection Main Street and Leisure Centre Driveway.....	39
Table 3.5: Model comparison in Intersection Ontario Street South and Childs Drive	40
Table 4.1: Summary of input variables.....	50
Table 4.2: Model comparison	53
Table 4.3: Forecasting performance for DeepAR model on hold-out intersections	60
Table 5.1: Prior-Known Future Time-Dependent Inputs.....	75
Table 5.2: Static Covariates	76
Table 5.3: Forecasting results of different models.....	78
Table 6.4: External covariates.....	97
Table 6.5: Comparison of forecasting models on test dataset.....	99
Table 6.6: Comparison of forecasting models on hold-out dataset (Few-shot scenarios)	100
Table 6.7: Comparison of forecasting models on zero-shot scenarios.....	100
Table 6.8: DTW distance for different models for different weathers.....	106

Chapter 1: Introduction

1.1 Background

Traffic congestion is a major concern worldwide, driven by urbanization, population growth, and economic expansion. For example, in the United States, traffic flow rose by 20% between 2002 and 2022 (FHWA, 2022), leading to roughly 54 hours of annual delay per commuter in 2019—up from 20 hours in 1982. Meanwhile, heavy traffic and congestion result in environmental pollution as commuters spend longer on the road (De Vlieger et al., 2000). Similar challenges exist in Canadian cities such as Toronto, where commuters face even longer delays than those in Los Angeles or New York (Spears & Kalinowski, 2010). In 2018, congestion in the Greater Toronto and Hamilton Area (GTHA) caused 81 hours of delay per commuter and CAD 3.3 billion in economic losses (Plan, 2016). Furthermore, these conditions can also result in driver frustration, aggressive maneuvers, and higher accident risk (Wang et al., 2013).

Traditionally, congestion is often addressed by building or expanding roads, but this approach faces challenges in funding limitations, environmental regulations, and land availability. Moreover, adding capacity can sometimes worsen congestion due to complex supply-demand factors (Daganzo, 1997). To overcome these issues, many agencies now rely on Intelligent Transportation Systems (ITS), especially Advanced Traffic Management and Information Systems (ATMIS). ATMIS uses real-time data and various technologies—such as adaptive traffic signals, automated incident response, and dynamic signage—to improve traffic flow, promote safety, and inform travelers (Dahlgren et al., 1996). Numerous North American cities have adopted or plan to adopt ATMIS. For example, Toronto aims to update adaptive signals using Big Data (Plan, 2016), while the U.S. Federal Highway Administration (FHWA, 2022) and major metropolitan areas leverage artificial intelligence platforms that utilize multi-source data for network-wide congestion detection, prediction, and remote signal adjustments to respond dynamically to real-time traffic demand shifts (Wang et al., 2019).

In line with these ATMIS initiatives, Turning Movement Flows (TMFs) are crucial to reducing congestion in urban road networks. By showing how traffic is distributed among all possible directions, TMFs enable ATMIS to optimize signal timing, allocate lane resources effectively, and coordinate phases across interconnected intersections, directly reducing congestion. Accurate forecasting of

intersection TMFs is therefore vital for realizing ATMIS’s potential. Predicting how vehicles will turn at key junctions helps agencies anticipate peak demand and respond proactively. For instance, during rush hours, ATMIS can rely on TMFs predictions to extend green lights for heavily used movements or synchronize signals into “green wave” corridors, minimizing stops and idling. With dependable TMFs data, traffic managers can adjust signals, deploy ramp metering, fine-tune dynamic signage, and introduce adaptive lane configurations or transit-priority measures. These strategies collectively reduce queue lengths, smooth out stop-and-go patterns, and shorten overall travel times—boosting both safety and mobility throughout the network. Focusing on TMFs also helps identify and relieve congestion hotspots, while improving road safety and overall performance.

However, timely, accurate and reliable forecasting of TMFs remains a challenging task due to the complex factors influencing urban travel demand—such as land use, driver behavior, and local signal policies—which vary significantly over space and time. Moreover, unexpected incidents like accidents, construction, major events, or adverse weather can trigger abrupt surges or drops in traffic flows, complicating predictions. These realities underscore the need for robust, flexible forecasting approaches capable of handling both normal fluctuations and sudden disruptions. While advancements from statistical models (Chikkakrishna et al., 2019; Yan, 2010) to modern deep learning architectures, e.g., LSTM (Gong et al.), Transformers (Xu et al., 2020), Graph Neural Networks (Yu et al., 2017), have expanded forecasting capabilities, key gaps persist in model interpretability, adaptability to new intersections, and domain-specific integration. To address these shortcomings, this research proposes a novel forecasting framework specifically designed for real-time TMFs prediction, incorporating external factors and explicitly supporting few-shot and zero-shot scenarios to enhance adaptability and real-world utility.

1.2 Research problem and objectives

In this research, TMFs forecasting problem can be defined as a typical multiple variable time series forecasting problem. TMFs data represents a series of values collected at different time interval. Consider a set of n_s intersections in an urban road network, denoted as S with each intersection, represented by s , where $s \in S$, having n_m movements. These intersections are instrumented, providing continuous traffic count data on all turning movement flows (TMFs). The collected TMFs dataset is organized in a time-series form at a fixed time interval (e.g., 5-minute); that is, it consists of $n_s \cdot n_t \cdot n_m$

values, representing the observed data for the n_m turning movements at n_s intersections over the past n_t time intervals. We can define the TMFs data for a specific intersection s as a multivariate time series as Equation (1.1).

$$X_{s,(1,t)} = \begin{pmatrix} x_{1,1} & x_{2,1} & \cdots & x_{t,1} \\ x_{1,2} & x_{2,2} & \cdots & x_{t,2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{1,m} & x_{2,m} & \cdots & x_{t,m} \end{pmatrix} \quad (1.1)$$

Where $x_{t,m}$ is the record of turning movement m at time t at intersection s (note that the subscript s is dropped for convenience).

The typical TMFs forecasting problem can be defined as using n_h intervals (input data range) of TMFs data from time interval $t - h$ to the time interval t to predict the future n_p intervals (forecasting horizon) of TMFs from interval $t + 1$ to $t + p$ of TMFs, as shown in Equation (1.2).

$$X_{s,(t+1,t+p)} = F(X_{s,(t-h,t)}) = F \begin{pmatrix} x_{t-h,1} & x_{t-h+1,1} & \cdots & x_{t,1} \\ x_{t-h,2} & x_{t-h+1,2} & \cdots & x_{t,2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{t-h,m} & x_{t-h+1,m} & \cdots & x_{t,m} \end{pmatrix} \quad (1.2)$$

Where $F(\cdot)$ represents the TMFs forecasting model, $X_{s,(t-h,t)}$ represents the input historical data of TMFs containing n_h historical values from interval $t - h$ to interval t , and $X_{s,(t+1,t+p)}$ represents the output future TMFs value from interval $t + 1$ to interval $t + p$.

In addition to the TMFs record data, the TMFs pattern can also be influenced by spatial features and other covariates. Therefore, it is necessary to include spatial features $V_{spatial}$ and covariates $V_{cov,t}$ as additional input data. Spatial features $V_{spatial}$ includes the locations of the intersection which are not varied by time, while $V_{cov,t}$ represents the features varied by the time, such as, expected weather, holidays, and scheduled events. Figure 3.1 illustrates the key aspects of the forecasting problem. In order to predict the next n_p intervals of TMFs data $X_{s,(t+1,t+p)}$, the forecasting model takes past TMFs data $X_{s,(t-h,t)}$, spatial features $V_{spatial}$, and covariates $V_{cov,(t-h,t)}$ as input and also benefits from considering

future observed covariates $V_{cov,(t+1,t+p)}$ as well as spatial features $V_{spatial}$, which remain constant over time.

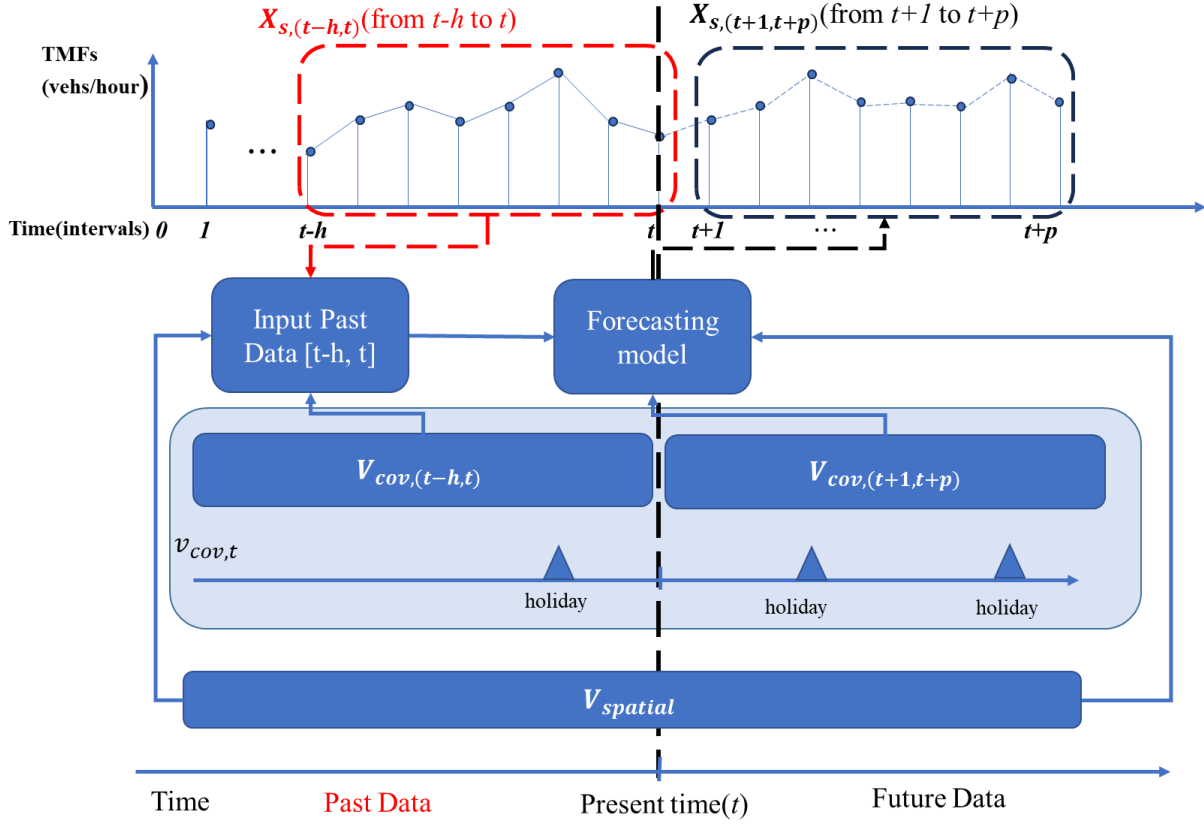


Figure 1.1: The TMFs forecasting problem

The ultimate goal is to create a robust mapping function capable of predicting real-time traffic flows at individual movement levels while accounting for the complex spatiotemporal dynamics inherent to urban intersections. To help achieve such goal, several specific objectives have been set for this research. They are:

1. To develop improved models for capturing complex spatial and temporal patterns
 - Develop models capable of learning short- and long-term dependencies in TMFs and enhancing generalization across different intersections and traffic patterns.
2. To quantify the impact of exogenous factors on TMFs forecasting accuracy

- Investigate how static (e.g., road type, speed zone) and time-dependent (e.g., weather, time of day) covariates influence forecasting accuracy and model reliability.
 - Evaluate the effectiveness of models that explicitly incorporate exogenous factors in improving predictive performance.
3. To design a robust framework for zero-shot and few-shot TMFs forecasting
- Develop a model architecture capable of generalizing to new intersections and cities without extensive retraining.
 - Test model adaptability under various forecasting horizons and weather conditions.
4. To evaluate the performance of hybrid deep learning models
- Compare the performance of encoder-decoder models (e.g., TFT, DeepAR) and decoder-only models (e.g., TimesFM) across different forecasting scenarios.
 - Assess model robustness and transferability using hold-out intersection tests, cross-city generalization, and weather-based scenario testing.

1.3 Thesis structure

This thesis follows a manuscript-based structure. Each of the main chapters (Chapter 3, 4, 5 and 6) consists of a manuscript published or submitted in a peer-reviewed conference or journal. The contribution of authors is presented in the statement of contributions in front matter.

Chapter 2 provides a literature review of the current state-of-the-art traffic forecasting models and studies related to TMFs forecasting. The first part of this chapter offers an overview of the statistical and machine learning models developed in previous studies for general road traffic volume forecasting, focusing on their primary features, data requirements, and performance. This is followed by a comprehensive review of past efforts dedicated specifically to the intersection TMFs forecasting problem, to identify the primary areas in need of improvement.

Chapter 3 contains an published manuscript (Zhang, Pan, & Fu, 2024). This chapter proposes a Parallel Bidirectional LSTM (PB-LSTM) model that combines stacked bidirectional LSTM (SBiLSTM) and a multilayer perceptron (MLP) to improve intersection Turning Movement Flows (TMFs) forecasting. The

model is trained and evaluated using turning movement counts (TMCs) from 22 intersections in Milton, Ontario, covering two years of data. Sensitivity analysis is conducted to assess the effect of input data range, forecasting horizon, and intersection clustering on model performance.

Chapter 4 contains an accepted manuscript (Zhang, Pan, Muresan and Fu (2024)). This chapter focuses on uncertainty quantification in TMFS forecasting using the DeepAR model, a probabilistic forecasting approach. The model is trained on 15 months of TMFs data from 39 intersections in Milton, Ontario and evaluated on forecast sharpness, reliability, and transferability. Experiments are conducted on hold-out intersections and different forecasting horizons to assess model robustness.

Chapter 5 contains an accepted manuscript (Zhang, Pan, Muresan, Lu, et al., 2024). This chapter investigates the effect of exogenous factors on TMFS forecasting using the Temporal Fusion Transformer (TFT) model. The model is trained on data from 76 intersections in Region of Waterloo over 507 days, incorporating 17 exogenous factors (e.g., weather conditions, road type, speed zone). Experiments examine the influence of static and time-dependent covariates on model performance under varying traffic patterns and forecasting horizons.

Chapter 6 contains a manuscript under review (Zhang et al., 2025). This chapter explores zero-shot and few-shot TMFs forecasting using the TimesFM model, a decoder-only approach. To enhance performance, the model incorporates Empirical Mode Decomposition (EMD) for feature extraction and Random Forest (RF) for integrating exogenous factors. The model is evaluated on 76 intersections from the Region of Waterloo for training and few-shot evaluation, and 36 intersections from Milton as zero-shot scenario. Additional experiments are conducted under extreme weather conditions (e.g., rain, snow) and across different forecasting horizons to assess model robustness and generalization.

Chapter 7 summarizes the main conclusions in this thesis, original scholarship contributions to knowledge, and recommendations for future research.

Chapter 2: Literature Review

There have been numerous recent studies devoted to applying statistics and deep learning models for addressing the challenge of traffic forecasting in general and intersection Turning Movement Flows (TMFs) in specific. This Chapter reviews the major past studies on traffic forecasting and state-of-the-art models in terms of their principles, data being used, model construct, applications, and limitations. It starts with an overview of the traditional statistical forecasting models, followed by an exploration of machine learning models and deep learning models developed in previous studies for time series forecasting and general road traffic volume forecasting. Lastly, the review focuses specifically on past research efforts dedicated to the problem of TMFs forecasting.

2.1 Traffic volume forecasting models

2.1.1 Statistical forecasting models

In earlier decades—when computing resources were still quite limited—statistical models played a pivotal role in traffic flow prediction. Linear regression models were one of the earliest models used in traffic volume forecasting (Smith et al., 2002) by exploring and quantifying the relationship between a dependent variable (traffic volume) and one or more independent variables (timestamps, location and other external factors). These models were shown some success in capturing the relationships between traffic volumes and various influencing factors such as time of day, day of the week, weather conditions, and land use (Dai et al., 2006; Smith et al., 2002). This type of regression models is easy to interpret but have limitations in capture trend and temporal correlations commonly existing in traffic data.

To better handle temporal patterns, the Autoregressive Integrated Moving Average (ARIMA) models have become the most popular classical statistical model for traffic time series forecasting (Zhang, 2003). ARIMA model uses past values of a time series to predict future values based on the regression of the historical data and the regression of the historical prediction error by performing Autoregressive and Moving Average, allowing the model to use information from the past instead of fixed influencing factors. In a study by Tong and Xue (Tong & Xue, 2008), ARIMA models were used to forecast one-year monthly traffic volume on urban highways in China based on the former 13 years. The authors compared the performance of ARIMA models with that of seasonal regression models and found that

ARIMA models provided more than 30% better results in terms of forecasting accuracy. Additionally, researchers have developed many advanced ARIMA models, such as seasonal ARIMA (Dong et al., 2009; Kumar & Vanajakshi, 2015; Williams & Hoel, 2003), ARIMAX (Johansen & Foss, 1993) and Space-Time ARIMA (ST-ARIMA) models, to capture seasonality and spatial patterns together.

Another notable statistical model is Prophet (Gibran & Bushrui, 2012), a decomposable time series model proposed by Facebook for time series forecasting. It decomposes the original time series data into a seasonal factor for recurrent features, a trend factor, a holiday component, and an error term (residual factor), which are then processed by different functions. In a study by Chikkakrishna et al. (Chikkakrishna et al., 2019), both seasonal ARIMA and Prophet models were developed using 6-day hourly traffic volume data collected from a highway segment, to forecast traffic flow for the next 24 hours. Their comparison showed that the Prophet model outperformed the seasonal ARIMA model slightly. Both models demonstrated satisfactory accuracy, exhibiting a percentage error of less than 2% in the small test dataset. These results align with findings from other studies that have explored the application of ARIMA and Prophet models in traffic volume forecasting, further reinforcing the effectiveness of these models. (Chikkakrishna et al., 2019; ChikkaKrishna et al., 2022; Wang et al., 2021).

While these models have shown promising results, they are not without limitations. Firstly, ARIMA and Prophet rely on the assumption that traffic flow data is stationary and has a linear relationship between input and output variables, which is often not the case. Secondly, most studies have found that these models are sensitive to outliers and can produce inaccurate results if there are extreme values in the previous interval. Finally, these models are often built on limited data and have not been tested in predicting long-term future trends and unexpected events.

2.1.2 Traditional machine learning forecasting models

The increasing availability of high-resolution traffic data has driven researchers to adopt machine learning methods for more accurate and dynamic traffic volume forecasting. These methods include K-Nearest Neighbours (K-NN) regression, Support Vector Regression (SVM), Random Forest (RF), and other ensemble learning models. Compared with statistical models, machine learning models can

effectively model the complex non-linear relationships in the massive data, and their structures are more easily combined with external features like weather conditions and road characteristics.

K-NN is a non-parametric supervised learning method that uses the similarity of datapoints to determine the forecasting data such as averaging the values of its k nearest neighbours. In a study by Smith and Demetsky (Smith & Demetsky, 1997), ARIMA was compared with K-NN for forecasting the next 15-minute traffic flow on two highways in Virginia, USA. The results showed that the average absolute error of K-NN was 18% lower than the ARIMA model. Another study by Zhang et al. (Zhang et al., 2013) used K-NN regression models to predict traffic flow in Shanghai, China, by analyzing 3 months of historical data collected from a traffic detection system that covered 21 ramps. One of the research objectives was to examine how temporal and spatial factors affected the performance of the K-NN models in predicting traffic volume. Their findings revealed that incorporating temporal and spatial features, such as time of day and ramp location, significantly improved the accuracy of urban freeway traffic flow forecasting for the next 5 minutes using K-NN, with an accuracy of over 90%. However, an obvious limitation of K-NN is that determining the hyperparameter k is difficult and computationally expensive, especially with long-term historical input data. Without the optimal hyperparameter k , the forecasting could be highly error prone. Meanwhile, the K-NN algorithm faces challenges in handling non-recurrent variations in data due to its reliance on the assumption that similar instances in the feature space yield similar outcomes. However, various non-recurrent events, such as traffic accidents or weather conditions, could introduce unique and unpredictable patterns, making it difficult for K-NN to accurately capture and predict these variations (Guo et al., 2003).

SVR was considered as another robust alternative for traffic forecasting before the advent of deep learning models (Cortes & Vapnik, 1995). The basic idea behind SVR is to map the input variables (features) into a higher-dimensional feature space using a kernel function. In this space, a linear regression model is trained using a subset of the training data called support vectors. The goal of the SVR algorithm is to minimize the training error while also maximizing the margin around the predicted values. Numerous studies have examined the effectiveness of SVR in traffic flow forecasting and consistently demonstrated its superior performance compared to K-NN. SVR surpasses KNN by effectively addressing temporal correlation in the data through the inclusion of lagged variables and time-related features in the input. By capturing the historical patterns and dependencies present in the data, SVR is able to model and predict traffic flow with higher accuracy and reliability (Castro-Neto et

al., 2009; Hu et al., 2016; Liang & Qiu, 2015; Luo et al., 2019). A study by Deshpande et al. (Deshpande & Bajaj, 2016) used SVR to make short-term traffic forecasts on a 6-day loop detector dataset in one mid-block segment. The results showed that the Root Mean Square Error (RMSE) of SVR for the next 1-minute aggregation interval traffic count prediction was only 0.42 vehicles per minute or 25 vehicles per hour. However, while SVR can handle non-linear data by mapping it to a higher-dimensional space, it may not be effective where the data is highly non-linear or has complex relationships between variables. This can lead to the overfitting problem, similar to the linear regression models.

In addition to K-NN and SVR, ensemble learning methods, including random forest (RF) and Adaptive Boosting (Adaboost), have also been explored for traffic forecasting. Instead of applying an individual machine learning model, ensemble models, which combine the predictions of multiple models to obtain a final prediction, were found to be more accurate and less prone to errors. RF models (Ho, 1995) are a type of ensemble learning algorithm that combines multiple decision trees (Quinlan, 1986) to improve the accuracy and stability of predictions. Liu et al. (Liu & Wu, 2017) developed an RF-based approach to forecast traffic flow and congestion on a busy urban road in Shanghai, China. Their dataset has 1124 independent records, and each record consists of different sources of data, including traffic data, weather data, and holidays, as input to improve the model generalizability. Their results showed that the congestion forecasting accuracy of the model was as high as 87.5%. Adaboost is another popular ensemble learning method (Freund & Schapire, 1997). It combines multiple “weak” models by sequentially training them on the same dataset and adjusting the weights of the misclassified samples. Different from RF that uses only decision trees, Adaboost can incorporate different elementary models as “weak” learners, such as linear regressors, SVR, and even RF. In a study by Leshem et al. (Leshem & Ritov, 2007), Adaboost was utilized to forecast the next 30 mins traffic flow on a highway based on 7 days of historical urban loop detector data. The results indicated that Adaboost was able to provide 60% better prediction performance than K-NN and SVR. However, ensemble models such as RF and Adaboost could be susceptible to overfitting. The combination of individual models in an ensemble increases the risk of memorizing the training data instead of capturing the underlying patterns, limiting their generalizing to unseen data. This overfitting issue arises when the ensemble model becomes overly complex and excessively relies on the idiosyncrasies of the training set, resulting in reduced performance on new and unseen data (Segal, 2004). Moreover, the process of creating subsets can be challenging and suboptimal hyperparameters can result in high bias and high errors in new datasets.

Traditional machine learning models are considered a milestone in solving forecasting problems, as they have shown a significant improvement in forecasting accuracy compared to traditional statistical models by effectively capturing complex nonlinear relationships and interactions within large and diverse datasets.

2.1.3 Deep learning forecasting models

Although traditional machine learning models have demonstrated promising results in various traffic volume forecasting studies, researchers are continuously developing advanced models that can improve prediction accuracy with larger and more complex datasets. With the latest advance in Artificial Intelligence (AI) and machine learning, various deep learning models have been successfully applied for solving many challenging engineering, science, and finance problems including those of forecasting. The main idea behind deep learning models (LeCun et al., 2015) is to create a multi-layered neural network, consisting of multiple layers of interconnected nonlinear transformation nodes or neurons, that can learn hierarchical representations of the input data. In these models, the first layer of the network receives the raw input data and applies a set of learned filters or features to extract low-level features. The output of the first layer is then passed to the second layer, which applies a different set of filters to extract higher-level features. The deep learning model benefits from the “deep” depth, which means a large number of layers, to capture complex non-linear relationships and data latent patterns. Its heterogeneous structure can be combined with a wide range of algorithms to adjust to different application needs.

Multilayer Perceptron (MLP) is the most typical deep learning model (Popescu et al., 2009). MLP consists of an input layer, one or more hidden layers, and an output layer. Each layer of MLP is composed of multiple artificial neurons, which process the input data using activation functions to produce an output as well. MLP has been applied in traffic engineering for solving the general traffic forecasting problem. In a study by Sate Innamaa (Innamaa, 2000), MLP was used to forecast the next 15 minutes of traffic flow and speed with the input of the previous 3 consecutive 5-minute flows and average speeds. The results showed that their three-layer MLP model had only a 9% mean relative error on 15-minute traffic flow predictions. Similarly, Kumar et al. (Kumar et al., 2013) used a three-layer MLP model to forecast the next 5 minutes of non-urban traffic flows during off-peak hours on a segment of India National Highway. The authors used various input variables such as day type, time of day, and

vehicle conditions to train the model. The results indicate that the mean square error of the model was less than 2.9 vehicles within a 5-minute interval, demonstrating that the MLP consistently and reliably predicts traffic flow during off-peak hours. However, MLP treat input time series data as independent and unordered, discarding the temporal order critical to time series. Rearranging the input sequence does not affect the output, violating the causal nature of time-dependent data.

The limitations of MLP led researchers to Recurrent Neural Networks (RNNs) (Sherstinsky, 2020), which incorporate hidden states and feedback loops to pass information from one time step to the next in temporal order. Unlike traditional feedforward neural networks, RNNs use a feedback loop that allows information to be passed from one step of the sequence to the next, making them capable of capturing the temporal dynamics and dependency of the data. In a comparison of different machine learning models for traffic time series forecasting, Van et al. (Van Lint et al., 2002) found that RNN models outperformed other forward machine learning algorithms, like MLP, by around 10% higher accuracy with 72% fewer parameters for next 1 min travel time forecasting on their simulation dataset. RNN is one of the most important revolutions in time series forecasting as it is the first model that was designed to model complex temporal dependency. However, although RNN can make use of temporal dependencies from time-series data, it suffers from the vanishing gradient problem, which means that the gradients of the loss function for the weights in early time steps become very small, making it difficult for the model to learn long-term dependency (Hochreiter et al., 2001).

To address the issue of the vanishing gradient problem in traditional RNNs, researchers have developed Long Short-Term Memory (LSTM) models that utilize memory cells and gates to learn and retain important information for an extended period, enabling the model to capture long-term dependency (Hochreiter & Schmidhuber, 1997). LSTM networks consist of memory cells that can store information for a prolonged period, input gates that control the flow of information into the memory cell, forget gates that control the retention of information in the memory cell, and output gates that control the flow of information out of the memory cell. They have been shown to have excellent ability for traffic forecasting (Fu et al., 2016; Tian & Pan, 2015). In LSTM, a memory cell maintains its state over time and includes explicit memory and gating units that regulate the information flow into and out of the memory. Fu et al. (Fu et al., 2016) employed an LSTM model to forecast the next 5 minutes of traffic flow in a highway tunnel using the previous 30 minutes of data. The authors compared the performance of the LSTM model with ARIMA and found that the LSTM model had a 12.5% lower

mean squared error in prediction accuracy compared to ARIMA. Similarly, Tian et al. (Tian & Pan, 2015) utilized an LSTM model to predict traffic flow for 15 mins (3 intervals), 30 mins (6 intervals), 45 mins (9 intervals) and 60 mins (12 intervals) into the future, using the previous 5 mins (1 interval) data. The study revealed that the LSTM model not only had the lowest errors compared to MLP and SVR, but also demonstrated high robustness for long-term multiple steps predictions (15 mins to 60 mins). However, while LSTM can capture longer temporal dependency, it is not well-suited for modelling periodic patterns that are commonly found in traffic volume data, such as daily and weekly patterns (Ran et al., 2019). Additionally, LSTM is not capable of capturing spatial dependency and the impact of external features that are typical for traffic forecasting problems.

To mitigate these shortcomings, many researchers have combined MLP, LSTM, and RNN elements into encoder–decoder architecture. This architecture consists of two distinct stages: the encoder, which processes historical data to extract meaningful representations and capture long-term dependencies, and the decoder, which leverages these representations to generate sequential forecasts. One notable example of an encoder–decoder model is the Temporal Fusion Transformer (TFT), which augments LSTM with an encoder–decoder attention mechanism (Vaswani et al., 2017). Several studies have explored the application of TFT in traffic volume forecasting, and their results have shown that TFT outperformed classical LSTM models by at least 10% in accuracy on their datasets (Cai et al., 2020; Grigsby et al., 2021; Yan et al., 2021). In addition to using previous interval data as input only, attempts have been made to integrate Convolutional Neural Networks (CNNs) into an LSTM model to account for additional features such as the number of lanes and occupancies in forecasting. In a study conducted by Ma et al. (Ma et al., 2020), they proposed a CNN-LSTM model to predict the next 5 mins (1 interval) traffic flow for each lane of 125 highway segments in the US. The prediction RMSE of the CNN-LSTM model is less than 10 vehicles per hour per lane. Other models, such as DeepAR (Salinas et al., 2020), have been developed and applied in traffic flow forecasting by combining additional constant observed variables such as weather, location, and intersection geography.

Moving beyond sequence-based only encoder-decoder models, many models have focused on projecting traffic data into graphs with nodes and edges and developing graph-based models for the special domain of traffic networks. These models include Graph Neural Network (GNN)(Hu et al., 2020; Scarselli et al., 2008; Xie et al., 2020), Graph Convolutional Network (GCN)(Z. Li et al., 2019; Yu et al., 2020; Zhao et al., 2019) and Spatial-Temporal Graph Convolutional Network (STGCN) (Pan et al., 2021; Yu et al., 2017). Graph-related models effectively utilize the graph structure to capture crucial attributes such as

distances, connectivity, and other properties of road segments within the traffic network. In contrast to time series forecasting models that rely on location factors as covariates, the results consistently demonstrate that graph-related models exhibit better spatial transferability (Pan et al., 2021; Yu et al., 2017).

While encoder–decoder approaches have proved effective in many cases, their reliance on extensive labeled data and task-specific fine-tuning has encouraged exploration of simpler, more flexible alternatives (Pourpanah et al., 2022). In parallel, decoder-only architectures have gained popularity, partly propelled by breakthroughs in large language models (LLMs). Unlike encoder-decoder models, decoder-only models eliminate the need for an explicit encoding phase, operating directly on time series data through autoregressive forecasting. Recent work shows that these models excel in time-series contexts. For example, Yong et al. (Liu et al., 2024) introduced AutoTimes, a decoder-only forecasting model based on LLMs, and demonstrated its strong performance in zero-shot traffic flow forecasting scenarios. Their results showed that AutoTimes, without fine-tuning, outperformed supervised encoder-decoder models like TFT and DeepAR, achieving an MAPE of 11.3% on M4 (Makridakis et al., 2020) dataset. Meanwhile, Abhimanyu et al. (Das et al., 2023) proposed the TimesFM (Das et al., 2023) which is the first decoder-only model specifically designed for time series forecasting, and their result showed TimesFM outperformed state-of-the-art decoder-only models including AutoTimes in zero-shot forecasting scenarios while using fewer parameters. However, these models have not yet been thoroughly tested under real-time traffic conditions and primarily focus on hourly or daily granularity. They also face substantial challenges, including difficulty incorporating external factors and high sensitivity to noise and short-term fluctuations—both of which are common in real-time TMFs forecasting scenarios.

Despite the wide array of deep learning approaches proposed for time series and traffic volume forecasting, relatively few have been validated in real-time intersection turning flow scenarios. In addition, many existing studies emphasize either temporal or spatial dependencies without considering other influential factors—such as weather conditions and roadway geometry—that drive diverse traffic patterns. These works generally rely on location-specific datasets of limited duration, leaving several key questions unresolved: How can the integration of external covariates (e.g., weather, intersection geometry, event-driven disruptions) improve forecasting accuracy, robustness, and generalizability of models? What are the critical factors influencing the generalization performance of traffic forecasting

models, and how can models be systematically evaluated for their transferability to new intersections and urban areas? What are the most effective forecasting models for addressing few-shot and zero-shot forecasting scenarios in forecasting, especially when historical data is scarce or unavailable?

2.2 Traffic Turning Movement Flows (TMFs) forecasting

Compared with the significant progress in general traffic volume forecasting models, the problem of intersection turning movement flow (TMFs) forecasting has received very limited attention. Due to the limited availability of large real-world datasets on TMFs, some early studies had to use simulation data to help train the models. Karnati et al. (Karnati et al., 2021) developed a simple three layers MLP model using Turning Movement Counts (TMCs) data from a SUMO simulation model. The result demonstrated that their model only had 0.001 mean squared error for predicting the next 375s TMCs in both oversaturated and undersaturated simulation conditions. Similarly, Li et al. (Li, Ban, et al., 2020) proposed a novel Deep Intersection Spatial-Temporal Network (DISTN) combining LSTM and CNNs models for TMFs forecasting based on VISSIM simulation data. Their study evaluated the performance of DISTN in 8 intersections with different combinations of external factors such as weekdays versus weekends and peak versus off-peak hours. By combining external factors such as weekdays and weekends, and peak and off-peak hours, the MAPE for 15-minute intervals decreased from 18.48% to 16.74%. While their studies demonstrated the performance advantage of deep learning models for TMFs forecasting, their findings have yet to be validated using real-world data. It is unclear what the performance of these models would be in capturing the various complexities and nuances of real-world traffic behaviour, especially under unpredictable non-recurrent events.

In contrast, Ghanim et al. (Ghanim & Shaaban, 2018) developed a MLP model using 4,175 hours (175 days) of turning movement data, collected from 691 four-leg and 156 three-leg intersections. Their findings demonstrated that the MLP model accurately predicted the next one hour turning movements (TMFs) with a weighted Root Mean Square Percentage Error (RMSPE) of 7.6%. Their study showed the performance of one hour interval TMFs forecasting, but they didn't to explore the model performance of multistep forecasting and smaller prediction intervals which are much more important for real time traffic management. Mahamoud et al. (Mahmoud et al., 2022) proposed an ensemble learning model to estimate the next cycle traffic turning movements in traffic signal cycle level, using real-time collected turning movement counts data and signal timing from 19 intersections along 2 corridors in Florida, USA. Their results showed that the ensemble model outperformed the linear

regression baseline models with MAPE 9.53% and 4.6% for through and left-turn movements, respectively. The single next cycle level forecasting can be used for dynamic traffic signal control, but their study failed to evaluate their model using hold-out data; as a result, it is unclear how their model would perform when applied to a new intersection. In order to improve the model's spatial transferability, a few studies tried to combine spatial impacts with a time series forecasting model. A study by Rahamn et al (Rahman et al., 2022) compared two deep learning architectures: Graph Convolutional LSTM (GCN-LSTM) and Graph Convolutional Encoder–Decoder LSTM (GCN-Encoder–Decoder) model to predict the next 4 h intersection-level traffic movement flow by 6-h input historical data sequence. In addition to the historical traffic turning movement flow data of 1 year, the authors also combined travel demand and land use data as input for training. Their results showed that the developed GCN-LSTM model outperforms all the other baseline models with 4.02 vehicles/hour, 59.37 vehicles/hour, and 2.47 vehicles /hour RMSE values for right turn, through, and left-turn movements in hold out test intersections, respectively. Their studies considered intersection spatial features but ignored the impacts of other covariates such as weather, intersection layout and special events. Also, restricted by the fixed aggregation interval of data, they cannot quantify the uncertainty of their predictions. In addition to machine learning approaches, statistical models have also demonstrated potential in short-term traffic movement forecasting, Li et al. (Li, Wang, et al., 2020) proposed a statistical regression model based on Partial Least Squares (PLS) for short-term traffic state prediction. They evaluated their model using 23 days of loop detector and GPS data collected from an intersection in Jinan, China. Their results indicated that the PLS method outperformed traditional forecasting techniques such as ARIMA, k-NN, and SVR, but its performance was slightly worse to advanced deep learning models, including LSTM and CNN, for 15-minute traffic turning movement flow forecasting.

Overall, few studies have investigated cutting-edge deep learning methods for TMFs forecasting, and those that have typically rely on datasets spanning less than a year or limited to a small set of intersections within a single region. This narrow scope limits the ability to comprehensively evaluate model performance across diverse scenarios, including various traffic conditions, events, and zero-shot scenario applications, as these studies frequently neglect critical external factors such as weather conditions and roadway geometry. Consequently, the generalizability and robustness of these models remain uncertain in real-world traffic conditions with varied demands or data scarcity.

2.3 Summary

In summary, researchers have made significant progress in developing time series forecasting models and advanced real-time traffic forecasting models with some of the models showing high levels of accuracy for the specific conditions being considered. However, despite this progress, there are still several knowledge gaps that need to be addressed:

First, there has been insufficient exploration of advanced forecasting models for TMFs applications. While some earlier neural network models such as Multilayer Perceptron (MLP) and Long Short-Term Memory (LSTM) have been widely investigated, some recent, more powerful deep learning models like DeepAR and TFT, which can capture complex temporal dependencies, remain underexplored in the field of traffic engineering. Similarly, emerging decoder-only frameworks such as TimesFM, designed explicitly for zero-shot forecasting scenarios, have not yet been extensively evaluated within the intersection TMFs context. Consequently, important questions about which state-of-the-art models best balance predictive accuracy, computational efficiency, and interpretability remain unanswered.

Second, the explicit integration of critical external factors influencing TMFs has been notably overlooked. External covariates—including weather conditions, intersection geometry, and event-driven disruptions—are crucial for accurately forecasting real-world traffic dynamics but are rarely incorporated systematically into existing models. For instance, adverse weather events like heavy rainfall or snowstorms significantly alter driver behavior and intersection throughput, yet few studies explicitly include such real-time weather data. Intersection-specific features, such as lane configurations, speed limits, number of lanes, and locations, also profoundly influence traffic flow dynamics but have seen limited explicit integration. Additionally, non-recurrent disruptions like accidents, construction activities, or special events, despite their considerable impact on TMFs, have been seldomly accounted for in current forecasting frameworks.

Third, the literature currently lacks adequate solutions for few-shot and zero-shot forecasting scenarios. Most existing models assume extensive historical data availability, limiting their applicability to new or sparsely monitored intersections. Few-shot learning scenarios—particularly relevant for newly instrumented intersections with limited historical data—have not been sufficiently explored. Similarly, zero-shot scenarios, involving entirely new locations lacking historical data, remain inadequately

studied. Additionally, cross-domain adaptability—such as adjusting predictions between weekday, weekend, and holiday traffic patterns—has not received comprehensive attention.

Finally, current evaluation methods for forecasting models often remain incomplete or biased, thereby undermining their practical utility. Evaluations commonly rely on coarse temporal resolutions (e.g., hourly or daily), neglecting finer granularities (e.g., 5 minute, 15-minute intervals) essential for real-time adaptive traffic signal control. Furthermore, existing validation approaches rarely incorporate robust uncertainty quantification or rigorously assess model performance under sudden congestion events, where substantial forecasting errors and temporal lags frequently occur, posing significant challenges for real-time traffic management.

Chapter 3: Real-Time Intersection Turning Movement Flow (TMFs) Forecasting using A Parallel Bidirectional Long Short-Term Memory (PB-LSTM) Neural Network Model

As discussed in the preceding chapters, real-time intersection turning movement flows (TMFs) forecasting plays a critical role for Advanced Traffic Management and Information Systems (ATMIS) in optimizing traffic efficiency and mitigating congestion. Our literature review has revealed significant limitations in existing forecasting methodologies, particularly their inability to address the intricate spatiotemporal dynamics of intersection-level traffic movements, as most models prioritize univariate traffic flow predictions rather than multivariate analysis.

To bridge these gaps, this chapter presents a novel hybrid framework that integrates a stacked bidirectional Long Short-Term Memory (LSTM) network with a multilayer perceptron (MLP). This architecture is engineered to capture both transient temporal fluctuations and cyclical trends embedded in TMFs data. A case study using leverage two years of turning movement counts (TMCs) data from 22 intersections in Milton, Ontario is conducted to validate the model's superiority over state-of-the-art methods in accuracy, robustness, and generalizability. Furthermore, the study systematically evaluates the framework's performance on unseen intersections, diverse forecasting horizons (short- to long-term), and varying input windows, demonstrating its adaptability to heterogeneous traffic conditions and operational reliability in real-world settings.

The remainder of this chapter proceeds as follows. The literature review section reviews classical, machine learning, and deep learning-based traffic prediction models. The methodology section details the proposed model, followed by the case study section with various experiments for sensitivity analysis. The results section discusses in the experiments and results. The main findings and directions for future research are discussed in the conclusion section.

3.1 Literature review

The problem of short-term traffic flow forecasting has been studied extensively in the field of transportation engineering. Many forecasting models have been developed, ranging from traditional statistical time-series models to the latest deep learning models. The most popular classical statistical model for time series prediction is ARIMA (Zhang, 2003). Researchers have applied ARIMA and its variants, such as seasonal ARIMA (Williams & Hoel, 2003) and ARIMAX (Johansen & Foss, 1993) models for traffic forecasting. Another statistical model, Prophet, is proposed by Facebook for time series forecasting. It is based on an additive model where non-linear trends are fit with annual, weekly, and daily seasonality, plus holiday effects. Chikkakrishna et al. (Chikkakrishna et al., 2019) have applied seasonal-ARIMA and Prophet models separately in short-term traffic forecasting. They found that a Prophet model can make use of unsmoothed data for traffic forecasting with lower and upper bounds.

Lun Zhang et al. (Zhang et al., 2013) applied K-nearest neighbors(KNN) for short-term traffic flow prediction and showed good performance results. Brian et al. (Smith & Demetsky, 1997) compared ARIMA with KNN and showed that KNN overperformed ARIMA based on average absolute error.

In addition to KNN, other machine learning algorithms such as support vector regression (SVR), random forest (RF), and traditional multilayer perception (MLP) have also been explored for traffic forecasting. Liu et al. (Liu & Wu, 2017) proposed an RF model to predict traffic flow and congestion. Their result showed that the accuracy of the model was as high as 87.5% in congested predictions. Deshpande et al. (Deshpande & Bajaj, 2016) conducted a performance analysis and found that SVR has superior performance compared to other time-series models, and its root mean square error (RMSE) for 1 min aggregation interval traffic count prediction is only 0.42 vehicle per min. Compared with a single regressor, Adaboost, as an ensemble learning model consisting of plentiful weak regressors, was shown to provide better forecasting performance, 60% better than KNN and SVR(Leshem & Ritov, 2007).

With the development of many machine learning techniques, the performance of traffic forecasting has improved significantly over the years. This is especially true with the latest deep learning models like convolutional neural networks (CNN), which are capable of dealing with a variety of large datasets(Ma et al., 2017; Mahdavian et al., 2021). Recently, the application of recurrent neural networks (RNNs) in

time series analysis has become dominant. RNNs are neural networks with hidden states which keep the previous time features for modeling sequential data. Van et al. (Van Lint et al., 2002) applied RNNs to model the freeway travel time, and their model was shown to outperform other forward machine learning algorithms by around 10% better with a 72% smaller model size. LSTM models, an improved variant of RNN with improved long-term and short-term temporal learning features, were shown to have an excellent ability for traffic forecasting (Fu et al., 2016; Tian & Pan, 2015). With the same input historical data length, LSTM has 37.5% less RMSE than the support vector regression (SVR) model (Fu et al., 2016; Tian & Pan, 2015). The LSTM model is recently extended to a new bidirectional LSTM (BiLSTM) to capture the forward and backward dependence in typical traffic time-series data, showing significant performance enhancement (Schuster & Paliwal, 1997). In addition, attempts have also been made to integrate CNN into an LSTM model to account for some additional features like number of lanes, occupancies in forecasting and their prediction RMSE around 10 vehicles per hour of all lanes (Ma et al., 2020).

Despite the significant progress in traffic forecasting in general, the problem of intersection turning movement forecasting has received limited attention. Most past efforts have focused on predicting turning movements using highly aggregated O-D data (Chen et al., 2012; Wu & Thnay, 2001). Yashaswi et al. (Karnati et al., 2021) developed a simple neural network-based model on turning movement count (TMCs) data from a SUMO simulation model and showed that a three-layer simple neural network-based model performed very well in both oversaturated and undersaturated conditions. Mohamed et al. (Ghanim & Shaaban, 2018) developed a Multilayer Perception (MLP) model, a fully connected class of feedforward artificial neural network (ANN) that generates a set of outputs from a set of inputs. Although their results showed their model performed exceptionally well, there are several limitations in their studies: (1) Their dataset resolution is low, and the dataset only covers peak hours, limiting the application of their model; (2) They treat the turning movement data as simple cross-sectional observations instead of time series, ignoring the potential autocorrelation and periodicity in the traffic data.

3.2 Methodology

3.2.1 Proposed model

In this study, the time series of D movements at an intersection is denoted as $X = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T]^T \in R^{T \times D}$ with T time steps, where $\mathbf{x}_t$ is a vector $\mathbf{x}_t = [x_t^1, x_t^2, \dots, x_t^D]$ denoting the movement flows at time t . The intersection movement flow prediction problem is defined as learning a function $F(\cdot)$, to map T historical TMFS to the TMFS at a specific prediction horizon, Δt , as expressed by Equation (3.1):

$$[\mathbf{x}_{T+\Delta t}] = F([\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T]) \quad (3.1)$$

As described in the previous section, many traffic flow forecasting models (i.e., $F(\cdot)$) have been developed in the literature, differing in fundamental model structure, assumptions on data availability, and application environments. Our proposed model, PB-LSTM, is a deep neural network model built upon two well-known neural network models, namely, long short-term memory (LSTM) and multilayer perception (MLP). The LSTM model (Graves, 2012) has been successfully applied for time series forecasting with long-term dependencies (Althelaya et al., 2018; Cui et al., 2020; Ma et al., 2020; Sharma et al., 2021). Over the past few years, many different forms of LSTM have been proposed for traffic flow forecasting; however, none has shown significant performance improvement over the basic LSTM model (Ma et al., 2020). The basic LSTM model consists of a sequence of processing cells of the same structure, as shown in Figure 1. The input of the LSTM cell includes: $\mathbf{x}_t$, the current variable vector; $\mathbf{h}_{t-1}$, the preceding output; and $\mathbf{C}_{t-1}$, the initial cell state. The cell state is the key to LSTM: it runs straight down the entire chain to keep the long-term information unchanged, with only minor linear interactions. When needed to remove or add information to the cell state, the LSTM provides three gates to protect and control the cell state. The first gate (Equation (3.2)) is the forgotten gate to decide what information to filter out from the previous one. It evaluates $\mathbf{h}_{t-1}$ and $\mathbf{x}_t$, and outputs a number between 0 and 1 for each number in the cell state $\mathbf{C}_{t-1}$. The next step is to decide what new information to store in the cell gate, as Equations (3.3) and (3.4) show. The input gate, including the sigmoid layer, determines which values to update, and a tanh layer creates a vector of new candidate values, $\tilde{\mathbf{C}}_t$. After that, the old cell state $\mathbf{C}_{t-1}$ will be updated into the new cell state $\mathbf{C}_t$ as Equation (3.5). In the last step, the sigmoid layer will decide what parts of the state to output, and then the output will pass through tanh and multiply the cell state as Equations (3.6) and (3.7).

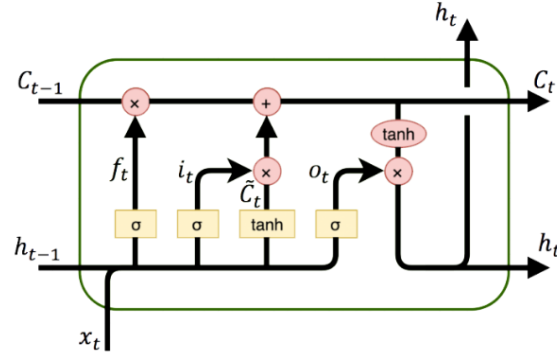


Figure 3.1: The structure of the basic LSTM cell (Sharma et al., 2021)

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (3.2)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (3.3)$$

$$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (3.4)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (3.5)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (3.6)$$

$$h_t = o_t * \tanh(C_t) \quad (3.7)$$

Multiple basic LSTM cells could be combined in a sequence to form new powerful unidirectional LSTM models. They can also be structured in two separate LSTMs which process a sequence of data in forwarding and backward directions, as shown in Figure 3.2. The resulting model - SBiLSTM has been shown to be substantially better than the unidirectional LSTM model (Cui et al., 2020). Furthermore, the SBiLSTM model can be built as a stacked multilayer form, resulting in a stacked SBiLSTM (Graves & Schmidhuber, 2005). In a stacked SBiLSTM architecture (Graves & Schmidhuber, 2005), the output of a

hidden layer will be fed as the input into the subsequent hidden layer, which has been found to be effective in dealing with certain types of tasks.

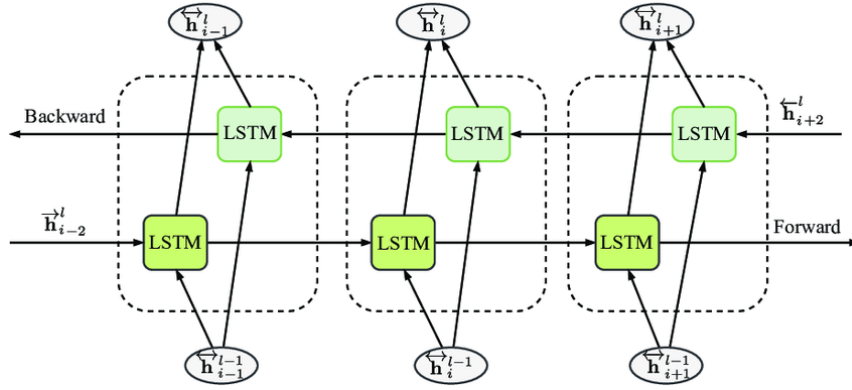


Figure 3.2: Stacked BiLSTM structure

In this study, an additional MLP layer is added to the top of the stacked SBiLSTM to capture the potential interactions between the individual movements at an intersection. The latter represents information with long-term relevance (memory). The resulting model is denoted as Parallel Bidirectional LSTM (PB-LSTM) model, as shown in Figure 3.3. This combined MLP and SBiLSTM structure is expected to make full use of both time-series and non-time-series data, enabling both short-term and mid-term forecasting. Each record includes two parts, X_{TMC} and X_{stamp} . In the upper part data, X_{TMC} , is TMCs time-series data for the D traffic movements, and each record indicates the aggregation count of 5 mins volume. This part of data is processed by the SBiLSTM block: the first layer of the SBiLSTM block is the time distributed dense layer which is used to capture the inner relationship of traffic movements for every temporal slice of input, and the process will not eliminate time dependence when capturing the TMCs dependencies; the subsequent three stacked layers are LSTM layers with 32 units, and they will analyze the time dependency for each movement. There is another feature of data X_{stamp} processed by an MLP block including hours of the day, day of the week, the month of the year, and intersection clustering label from the KNN model (which will be described in the later section); the MLP block consists of three dense layers with each having 16 units with Relu (Agarap, 2018) activation function. After the two blocks, the 1-D output vectors from X_{TMC} and X_{stamp} are concatenated together and passed to the dense layer to make the final prediction.

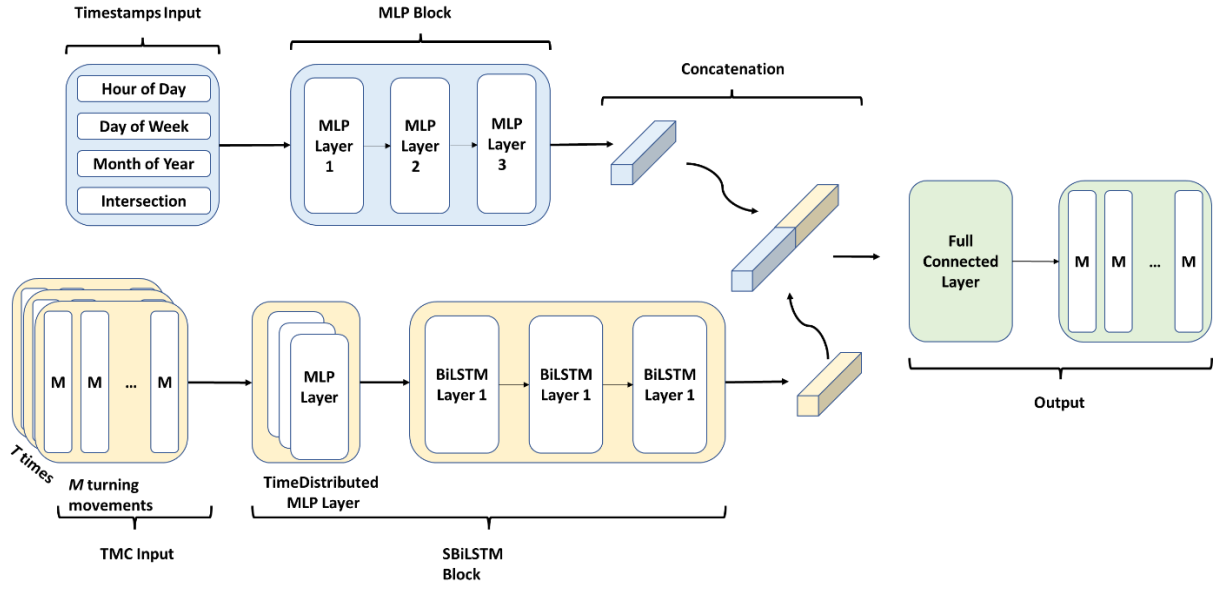


Figure 3.3: Model Summary

3.2.2 Case study data description

A case study is conducted to evaluate the performance of the proposed PB-LSTM model in comparison to several existing state-of-the-art models. The dataset used in the study includes TMCs at 5-min intervals at 49 signalized intersections in Milton, Ontario, Canada, for the period from 2019 to 2021 (Figure 4). The TMCs were collected using the Miovision SmartSense system with the main features summarized in Table 3.1.

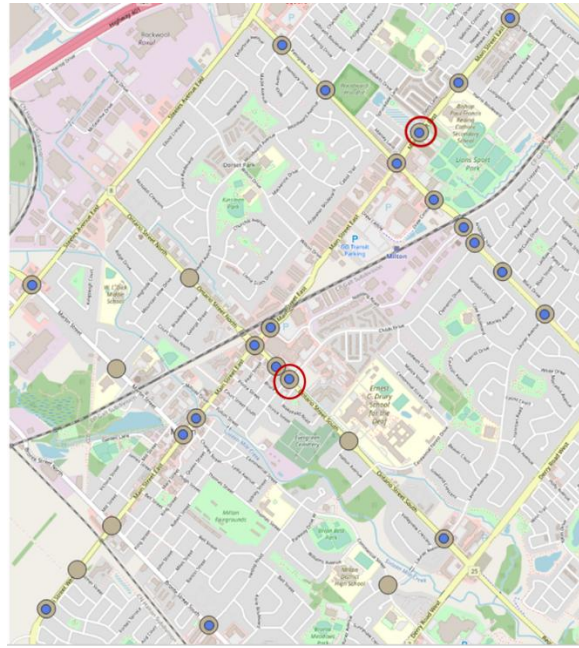


Figure 3.4: Case Study Intersections (The brown dots are the intersections being monitored, the blue dots are the intersections selected for analysis, and the intersections in the red circle are additional test intersections)

Table 3.1: The features included in the raw data

Type	Features		
Movement	TH (Through Traffic)	LT (Left Turn plus U-turn)	RT (Right Turn)
Counts	Counts of each record		
Location	Latitude	Longitude	
Time	Hour of Day	Day of Week	Month of Year

In this study, 20 intersections were selected that have the most complete data coverage from January 2020 to April 2022. The dataset has a total of 4,901,760 records or 245,088 records per intersection. The dataset is split randomly into two subsets: 80% for model training and 20% for model testing.

An exploratory data analysis was conducted on an example intersection (Main Street East and Maple / Sinclair). As a typical time-series dataset, TMCs records have daily and weekly seasonality and monthly tendency. Figure 3.5 shows the total 5-min volume of the Eastbound approach daily count from January 2020 to April 2022. From Figure 3.5, it is clear to see the trend over the year: the traffic has the first peak at the end of March by continuously growing from January, but then it starts falling until May;

after May, the TMC kept increasing to the second peak in September and then decreasing slightly in 2020 or being stable in 2021 until December. Compared with 2021 and 2022, these noticeable variations in traffic in 2020 were caused by a provincial-wide lockdown in Ontario during these months due to the COVID-19 pandemic, such as rapidly decreasing in April and slightly decreasing in October.

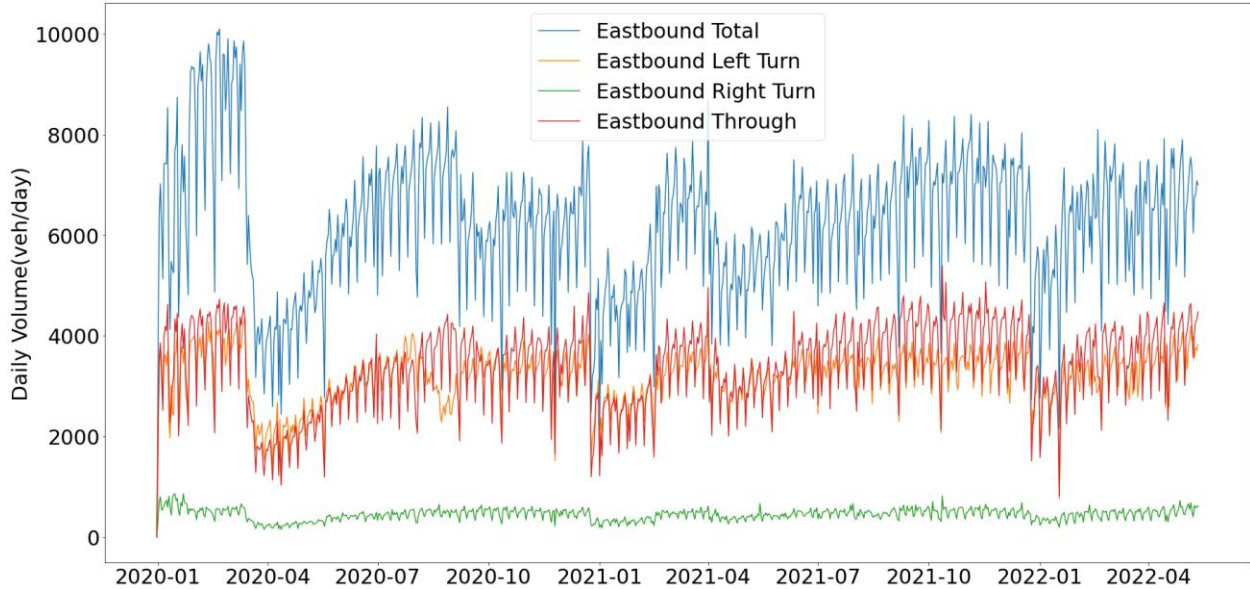


Figure 3.5: Daily TMCs over the analysis period

This data analysis has shown the influence of the COVID-19 pandemic on TMCs and confirmed the expected shorter temporary variation patterns in the TMFS, such as daily and weekly seasonality. To learn these time-varying patterns, it is necessary to apply models that can capture both short-term and long-term traffic dynamics, such as the proposed PB-LSTM.

3.3 Computational experiments and results

The proposed PB-LSTM model along with several base models described in the previous section was implemented using the Keras backend with TensorFlow (Ketkar, 2017). The deep neural networks were trained by optimizing the cost function using the Adam method (Kingma & Ba, 2014). In the training process, the training dataset was first re-formatted for models to include input history of a specific length. Cross-validation was applied at every iteration using 20% of the training dataset selected randomly for model validation. Model parallelism was applied in the computation process: the LSTM

and MLP blocks are trained in different GPUs and then concatenated on the CPU. The model hyperparameters have been decided by Bayesian optimization. The average training time for one-step forecasting is 52 seconds per iteration on two NVIDIA GeForce GPUs (GTX1080 8GB & GTX1070 8GB). The forecasting on the test dataset was completed within 90s, which implies that each prediction took place in under $0.65 \times 10^{-3}s$. The following section discusses the main results of several computational experiments.

3.3.1. Preliminary performance evaluation

To provide a basis for assessment and comparison, we first implemented several of the most well-known models, including ARIMA (J. D. Hamilton, 2020), Prophet (Taylor & Letham, 2018), Linear Regression (Yan & Su, 2009), AdaBoost (Freund & Schapire, 1997), Random Forest (Biau & Scornet, 2016), Multilayer Perception (MLP) (McCulloch & Pitts, 1943) and LSTM (Hochreiter & Schmidhuber, 1997) and compared their relative performance. As a preliminary evaluation effort, all models were trained and tested using one year of data (2020) at a given intersection (Main Street East and Maple / Sinclair Drive). The input time-series data range was set to 30 mins (i.e., previous six 5 mins time intervals) and the forecasting horizon 5 mins (i.e., next time interval). The performances of the models were evaluated using the average Root-Mean-Square-Error (RMSE) for all movements at the intersection.

Figure 3.6 shows the RMSE for the forecasted TMFS in terms of 5-min traffic volume. For MLP, RF, and Adaboost, the input is temporary variation patterns including the hour of the day, the day of the week, and the month of the year, while LSTM, ARIMA, and Prophet took the same range of observations as input. Figure 3.6 shows that among all models LSTM had the best performance, followed by MLP, AdaBoost and RF, which performed slightly better than the classical models. These results are consistent with the findings from the literature (Ma et al., 2020).

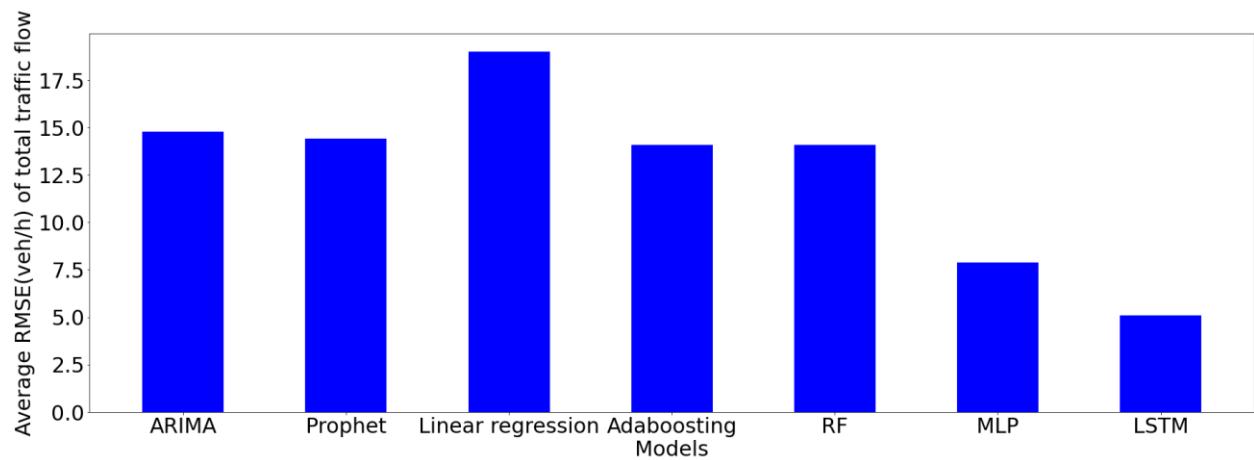


Figure 3.6: A comparison of performance of some well-known time-series forecasting models

Another preliminary computational analysis was conducted to compare the performance of three types of LSTM models, namely, basic LSTM, SBiLSTM, and SBiLSTM with an additional time-distributed full connected perception layer (MLP+SBiLSM). By combining with an extra MLP layer, SBiLSTM is expected to enhance the forecasting accuracy (Kowsher et al., 2021). These models were trained using data from the same test intersection and the performance results are shown in Figure 3.7. It can be observed that SBiLSTM and MLP+SBiLSTM significantly overperformed the basic LSTM, especially in weighted average RMSE. The performance of SBiLSTM and MLP+SBiLSTM are relatively close in terms of average RMSE; however, the RMSE of MLP+SBiLSTM for left and right turn movements is lower than SBiLSTM. As a result, MLP+SBiLSTM was finally selected for further consideration.

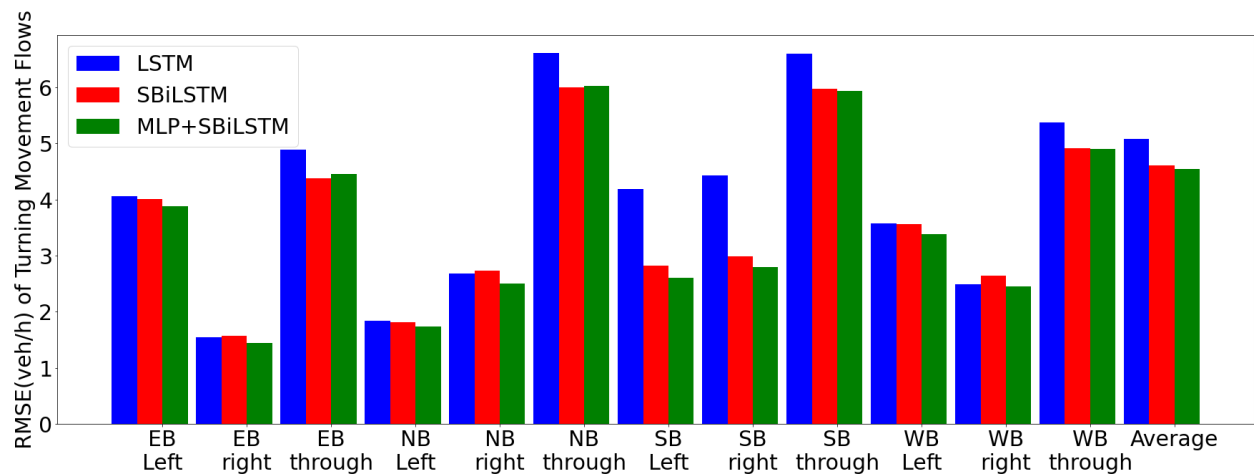


Figure 3.7: Comparison of Turning Movements Forecasting Errors by LSTM, SBiLSTM and SBiLSTM with MLP

The preliminary computational study has shown that both MLP and SBiLSTM models can outperform other models and therefore could be used as a building block for constructing new forecasting models such as the proposed PB-LSTM described previously.

3.3.2. Overall performance

The intersection of Main Street East and Maple/Sinclair Drive is selected for a case study on the overall performance of the PB-LSTM model, with TMCs data covering the period from January 2020 to April 2022. In this experiment, the input data range is 30 mins (i.e., six 5-min intervals) and the forecasting horizon is 5 minutes, i.e., one interval into the future, which was done in a rolling horizon scheme.

Figure 3.8 shows the forecasted intersection daily flows of Eastbound by turning movement versus the observed for 2020 during the COVID-19 pandemic. In March, the Ontario Government announced the lockdown for the first time, so there was a sharp decline. Our model successfully captured the general trend of traffic variation over the period.

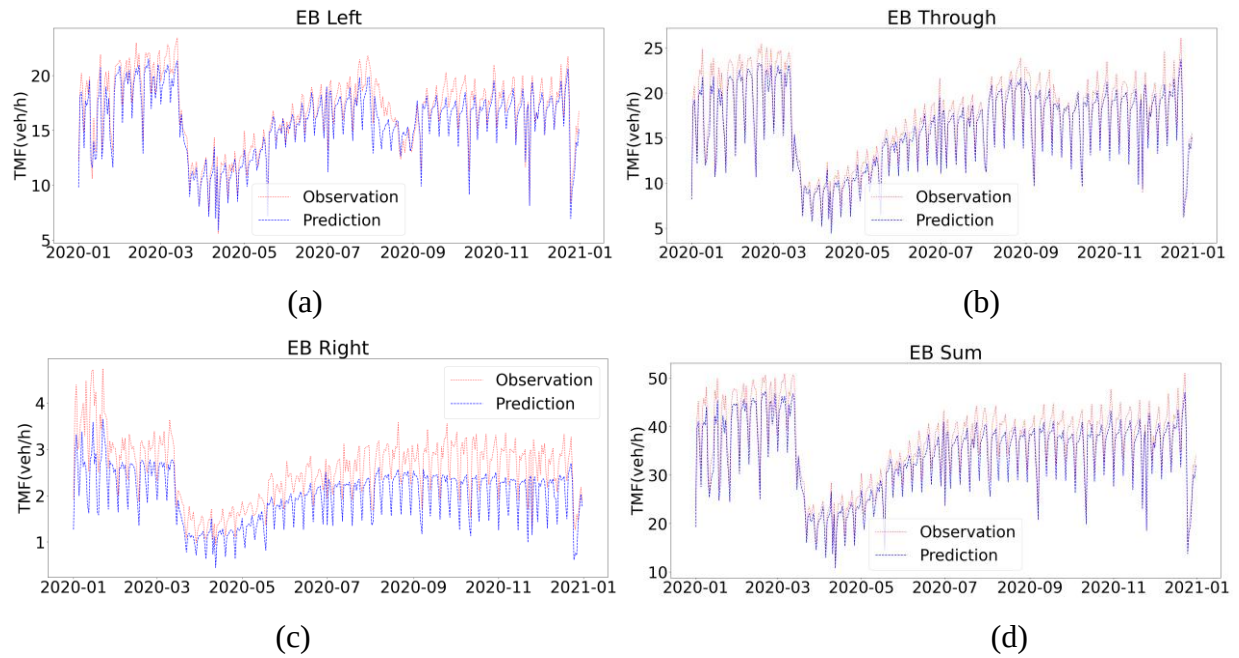


Figure 3.8: Performance of the PB-LSTM model for forecasting turning movement flows: predicted vs. observed daily volume (From 2020 Jan 01 to 2020 Dec 31, 30 mins input data range and 5 mins forecasting horizon)

Figure 3.9 shows the forecasted intersection flows by turning movement versus the observed for weekdays and weekends on the testing dataset for the months from January to December of 2020, respectively.

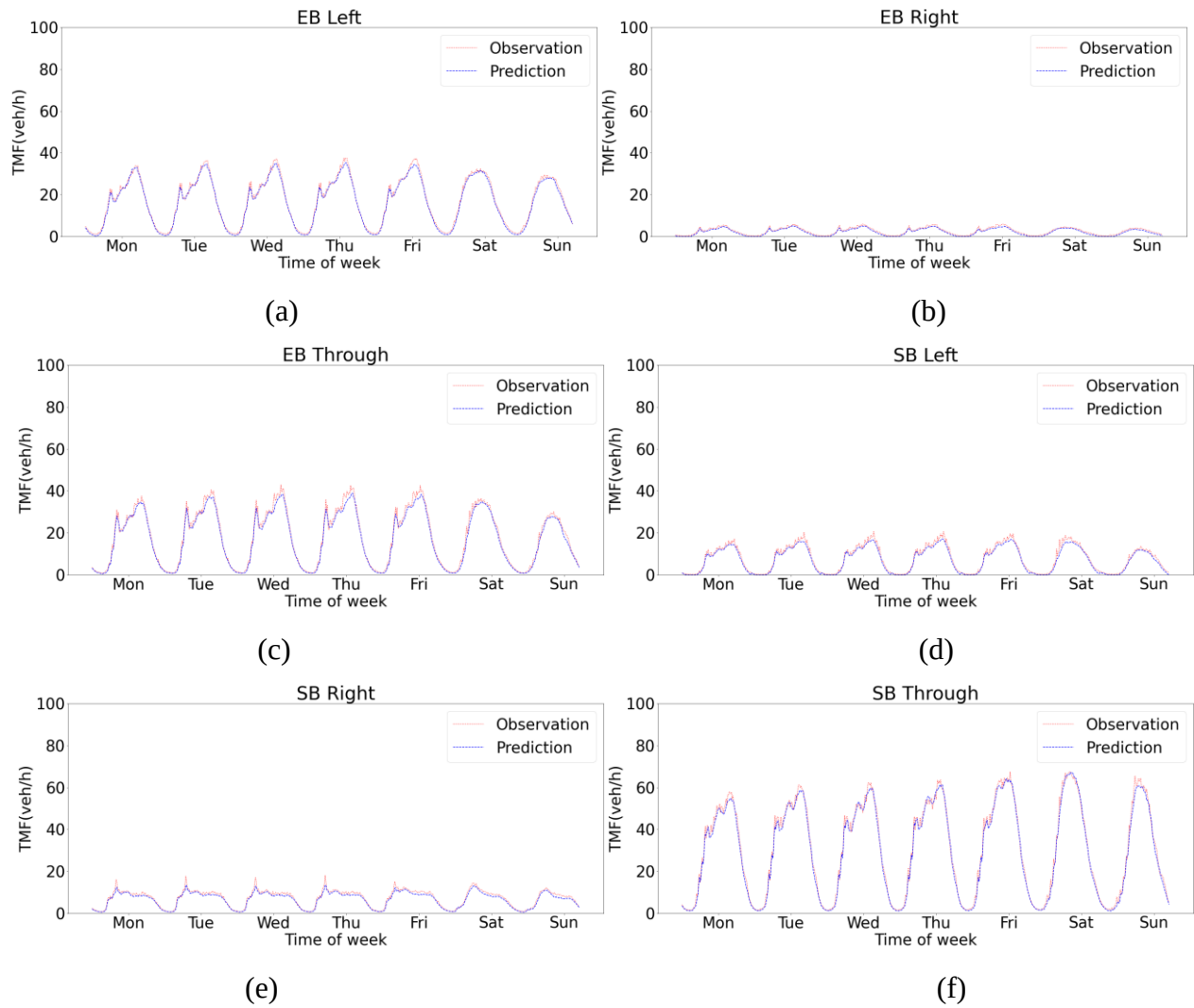


Figure 3.9: Performance of the PB-LSTM Model for forecasting turning movement flows: Predicted vs. Observed 5-min volume (weekdays and weekends, 30 mins input data range and 5 mins forecasting horizon)

It can be observed that our proposed model was able to accurately capture the particular daily variation of the traffic at this intersection. This could be attributed to the integration of the MLP and SBiLSTM models with the former helping acquire longer memory (i.e., the daily periodicity) and the latter gaining shorter memory on the time-of-day traffic variation. The result shows that the PB-LSTM model was successful in replicating the peaking of most traffic turning movements. However, the forecasting errors appeared to be larger during the peaking points, which is most likely due to the inherent random variation of traffic, especially when a short time interval (e.g., < 5 mins) is considered. Compared with weekday prediction, TMFS during the weekend only has one peak and the forecasting errors of the PB-LSTM model are also much smaller.

From Figure 3.8 and Figure 3.9, it can be observed the PB-LSTM can correctly forecast the trend, but the predicted counts are lower than the actual counts during the high flow conditions. Figure 10 shows the variation of Root Mean Square Error (RMSE) by the time of the day of the PB-LSTM model and SBiLSTM for eastbound of the same intersection on the testing dataset for the months from January to December 2020. It can be observed that our proposed model outperforms SBiLSTM most of the time for all movements. However, the RMSE are generally in proportion to the magnitude of the traffic flow being forecasted.

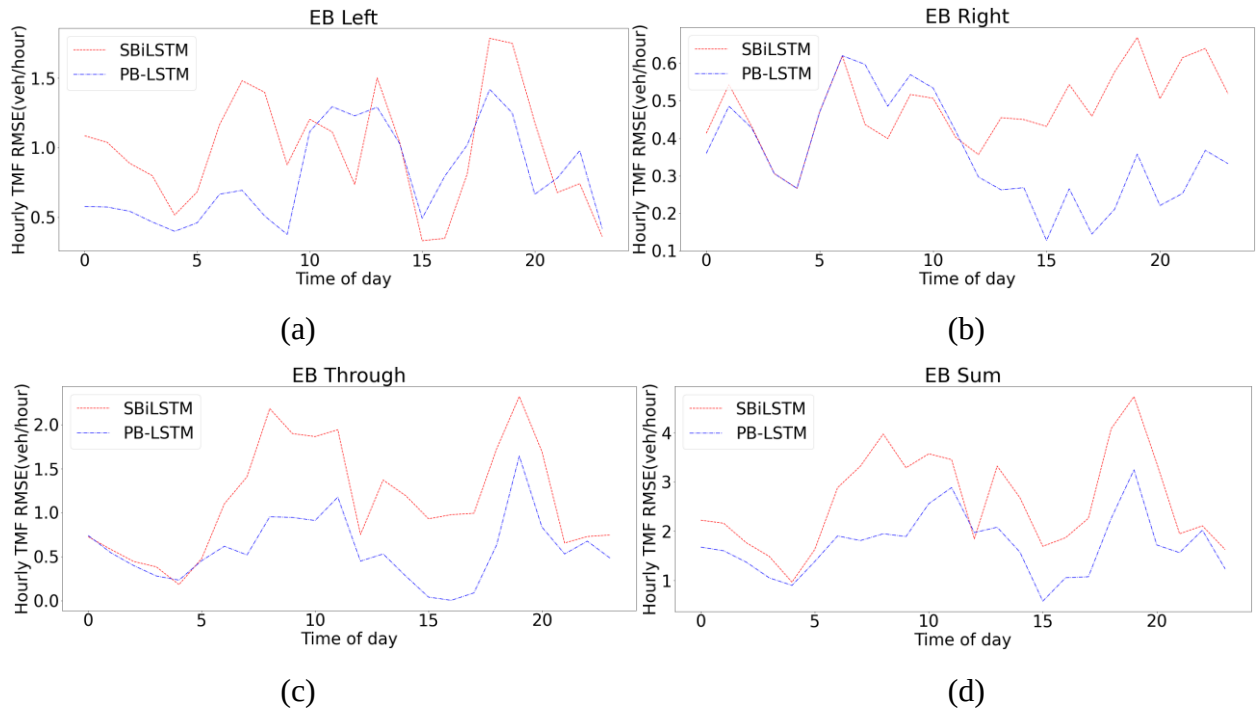


Figure 3.10: The average RMSE of the PB-LSTM Model and SBiLSTM model for forecasting turning movement flows in eastbound (30 mins input data range and 5 mins forecasting horizon)

Table 3.2 shows the performance of our proposed model on 20 intersections for all 12 intersections. It is obvious that the prediction error of PB-LSTM is acceptable for all test intersections.

Table 3.2: The RMSE of PB-LSTM for Forecasting Turning Movement Flows of 20 Intersections

	Eastbound			Northbound			Southbound			Westbound			Weighted Average
	Left	Right	Through	Left	Right	Through	Left	Right	Through	Left	Right	Through	
1	1.39	0.92	3.36	0.71	0.41	0.55	1.66	1.59	0.60	0.61	1.41	3.80	2.89
2	0.04	0.03	0.03	0.11	1.58	6.22	1.56	0.13	6.27	1.61	1.61	0.05	5.49
3	1.18	1.77	1.14	1.53	1.67	5.32	1.14	1.34	5.39	1.70	1.19	1.31	4.13
4	0.54	2.33	0.53	2.29	1.90	5.99	0.66	0.63	5.72	1.98	0.56	0.63	4.87
5	2.74	1.00	3.30	1.25	1.70	4.35	1.90	1.87	4.39	2.41	1.65	3.66	3.32
6	1.88	0.04	2.50	0.02	0.02	0.05	0.84	1.96	0.03	0.04	0.97	2.72	2.26
7	0.46	1.04	0.41	1.09	0.82	4.86	0.93	0.57	5.66	0.71	0.97	0.54	4.68
8	0.82	0.05	4.26	0.02	0.02	0.03	0.89	0.71	0.03	0.04	0.79	4.74	4.32
9	2.19	1.95	0.58	1.74	0.55	4.91	0.40	2.09	5.24	0.36	0.63	0.47	4.12
10	1.59	0.06	4.29	0.03	0.03	0.02	1.58	1.55	0.02	0.03	1.78	4.65	3.75
11	0.03	2.07	3.11	2.55	2.20	0.06	0.03	0.04	0.06	2.34	0.03	3.26	2.74
12	0.98	0.74	0.88	0.66	0.95	4.24	1.18	1.03	4.79	1.06	1.27	0.96	3.73
13	2.50	2.24	0.91	2.45	0.81	3.58	0.52	2.32	4.21	0.70	0.53	0.67	3.05
14	1.75	0.08	3.37	3.90	2.84	3.33	1.03	1.60	3.32	3.09	1.22	3.34	3.13
15	1.29	2.22	0.02	1.99	0.09	6.11	0.06	1.28	6.39	0.06	0.04	0.05	5.43
16	0.88	0.78	3.69	0.85	1.10	0.71	1.41	0.78	0.79	1.07	1.24	4.07	3.16
17	2.11	1.85	3.42	1.95	1.30	5.30	1.19	1.97	6.08	1.25	1.14	2.72	4.35
18	3.14	3.05	5.14	1.94	2.37	5.51	2.33	3.24	4.50	1.81	2.10	4.79	4.24
19	0.12	0.97	2.64	1.02	0.89	0.18	0.42	0.16	0.19	0.79	0.22	2.80	2.48
20	4.53	5.33	4.05	2.89	1.50	1.20	1.46	2.59	1.40	1.63	1.50	4.45	3.79

3.3.3. Effect of input data range

In this experiment, the proposed model along with the two base models (SBiLSTM and MLP) are compared for predicting the next 5 mins of TMFS under different input data ranges (past 5 mins, 10 mins, 30 mins, 60 mins, and 120 mins of observations) using data from the intersection of Main Street and Leisure Centre Driveway.

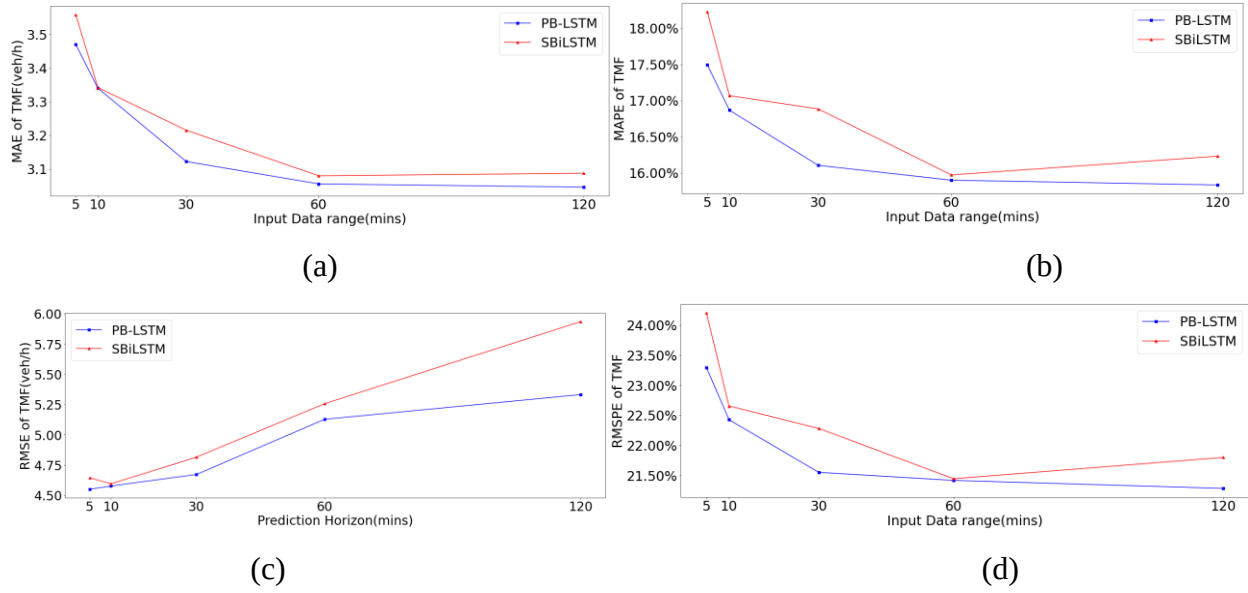


Figure 3.11: The weighted average performance comparison of 2 models under different observation range when predicting the next 5 mins flow

Figure 3.11 shows how the performance of the SBiLSTM and PB-LSTM models, as measured by four error measures (Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), Root Mean Square Percentage Error (RMSPE)), varied by the length of input data. Note that the result of the MLP model was not included as its errors are an order of magnitude larger. The forecasting errors for both models decreased quickly when the input data range increased from 5 mins to 30 mins and then leveled off. It also shows that the proposed PB-LSTM needed a shorter length of input data to achieve most of the performance gain compared to the Bi-LSTM model. The optimal input data length could be as low as 30 mins for the PB-LSTM model and 60 mins for the SBiLSTM model. It should also be noted that the proposed PB-LSTM model outperformed the SBiLSTM model consistently under all input data ranges.

3.3.4. Effect of the forecasting horizon

The performance of time-series forecasting models typically deteriorates as the forecasting horizon increases. The focus of this section is, therefore, to examine the underlying performance pattern of the proposed model with the objective of quantifying the speed and magnitude of the deterioration in comparison to the base model. The trained model was applied in a rolling horizon to forecast the future traffic movements at the same case study intersection as in the previous experiment with a forecasting

horizon ranging from one interval (5 mins) to 24 intervals (120 mins). As expected, Figure 3.12 shows that the forecasting errors of both models increased approximately linearly by the forecasting horizon. However, the proposed model showed a clear performance advantage over the SBiLSTM model both in terms of error magnitude and performance deterioration speed.

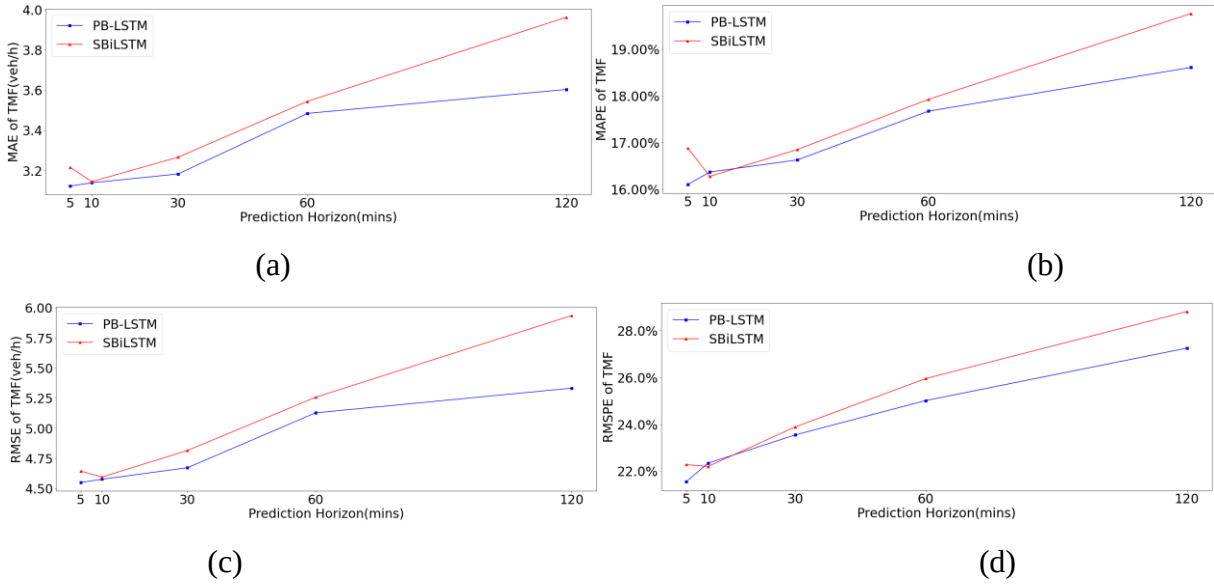


Figure 3.12: The weighted average performance comparison of 3 models for different prediction horizons when inputting 30 mins historical TMFS.

3.3.5. Benefits of location clustering and model transferability

Traffic flows at some intersections in the same area could exhibit similar time-varying patterns due to the similarity in the underlying travel demand and behavior. It is hypothesized that if clusters of intersections with similar attributes and traffic dynamics could be identified in advance and separate models be constructed for each cluster, these models could provide better forecasts, especially when the models are applied to new locations (intersections) of which observations are not available for training the models. In order to test this hypothesis, the 20 intersections were classified into four groups by the well-known K-means algorithm (Fix & Hodges, 1989) based on three different sets of intersection features as shown in Table 3.3.

Table 3.3: K-means classification scenarios

Input Features						
Scenario 1	Latitude	Longitude	Number of lanes for EW	Number of lanes for NS	Exclusive left lane for EW	Exclusive left lane for NS
Scenario 2	Daily TMFS for each approach					
Scenario 3	Daily TMFS for each movement					



Figure 3.13: Classification results (scenario 1: left, scenario 2: middle, scenario 3: right)

Figure 3.13 shows the classification results under the three scenarios. From the result, it is clear that the classification under scenario 2 and scenario 3 are almost the same, indicating the approach daily volume and the turning movement daily volume input showed similar traffic variation patterns. As a result, only scenario 1 and scenario 2 are considered in further analysis.

Two separate PB-LSTM models with intersection cluster indicators from the two clustering scenarios as an additional input were developed, namely, PB-LSTM Geo (Scenario 1) and PB-LSTM App (Scenario 2). The base PB-LSTM is one with no clustering but only intersection IDs, as denoted by the PB-LSTM ID. The input data range was set for 30 mins and the forecasting horizon for 5 mins.

The performance results are shown in Figure 3.14, Two observations can be made from these results. First, the two models considering clustering performed better than the one without considering clustering. Secondly, the performance superiority of the two models with different clustering scenarios

is mixed with no consistent outcome at some intersections, the PB-LSTM Geo outperformed PB-LSTM App while at the others the opposite was true.

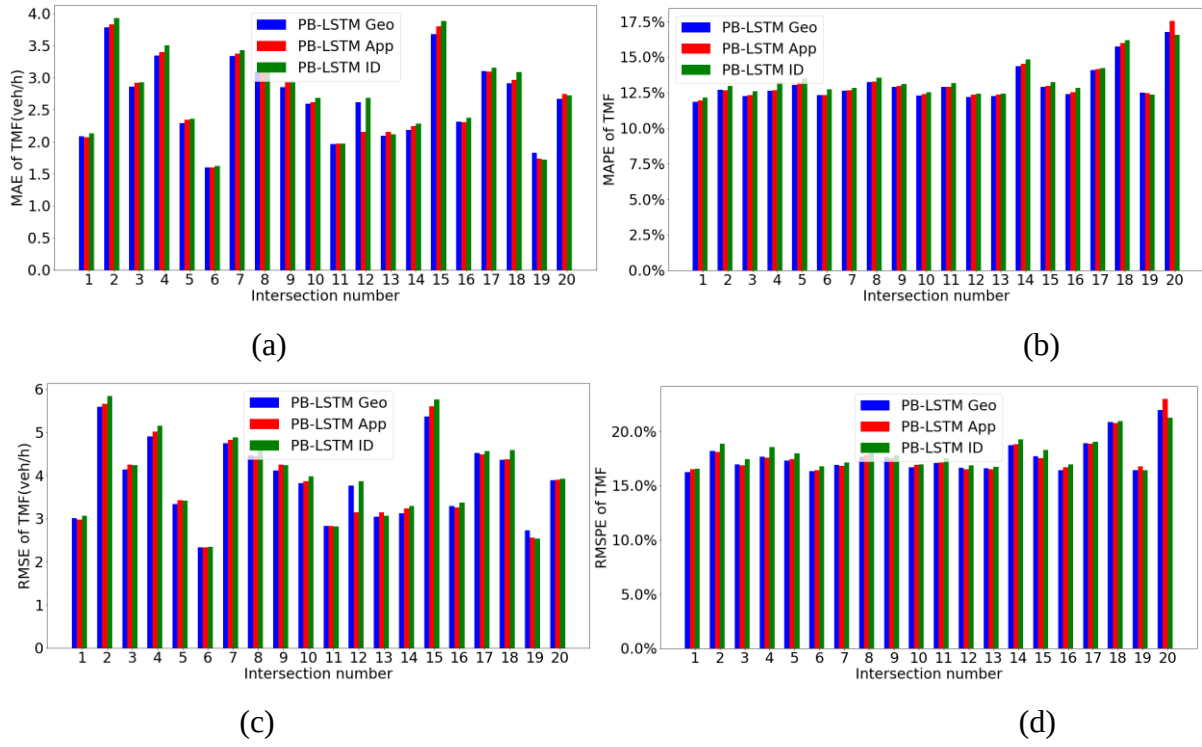


Figure 3.14: Comparison of PB-LSTM with Geometry classification label, PB-LSTM with approach classification label, PB-LSTM with ID numbers, and typical LSTM models when applied in 20 intersections

One of the main reasons for including a clustering step is to improve the transferability of a forecasting model, that is, the ability of a model to forecast the flows at a completely new location or intersection. The two trained models were applied to two completely new intersections – of which data were not used in training the proposed model: the intersection of Main Street and Leisure Centre Driveway (Geometry label:2; Approach label:2) and the intersection of Ontario Street South and Childs Drive (Geometry label:3; Approach label:2). The trained models were applied to generate one step ahead forecasts of the turning movements at the test intersection. The cluster of the test intersection was determined according to each clustering method. For simplification, this experiment considers only RMSE and RMSPE for performance evaluation. The input data range was fixed at 30 minutes. The result for Eastbound and Northbound is shown in Figure 3.15&3.16, and other details are in Table 3.4 & 3.5.

For forecasting the turning movements at the new intersections, the PB-LSTM Geo performed better for movements of high traffic volume but not for low traffic movements. Regardless of the clustering step, the forecasting errors of the proposed models for the new intersections (data were not used in training the models) are comparable to those for the existing intersections (data were used in training the models).

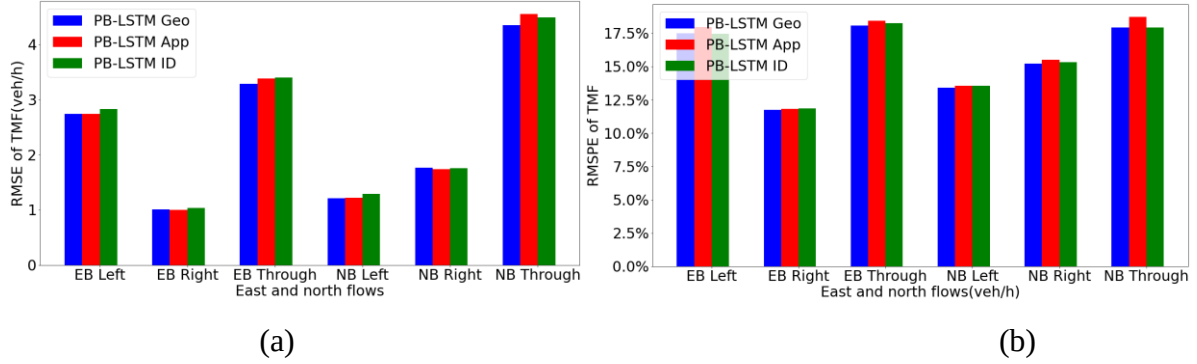


Figure 3.15: Comparison of PB-LSTM with Geometry classification label, PB-LSTM with approach classification label, PB-LSTM with ID numbers, and typical LSTM models when applied in Ontario Street South and Childs Drive.

Table 3.4: Model comparison in Intersection Main Street and Leisure Centre Driveway

		Eastbound			Northbound			Southbound			Westbound		
Turning Movements		Left	Right	Through	Left	Right	Through	Left	Right	Through	Left	Right	Through
Traffic Volume		1549910	216538	1708238	314000	668618	3146787	737520	508034	2882727	1206939	589729	2154083
RMSE	PB-LSTM(Geo)	2.74	1.01	3.28	1.21	1.77	4.35	1.93	1.78	4.41	2.43	1.73	3.68
	PB-LSTM(App)	2.74	1.00	3.39	1.22	1.74	4.55	1.92	1.77	4.51	2.47	1.69	3.81
	PB-LSTM(ID)	2.82	1.03	3.40	1.29	1.75	4.49	1.96	1.93	4.53	2.49	1.70	3.78
RMSPE	PB-LSTM(Geo)	17.5%	11.8%	18.1%	13.4%	15.2%	17.9%	15.4%	16.2%	18.7%	15.8%	15.0%	17.8%
	PB-LSTM(App)	18.0%	11.8%	18.4%	13.6%	15.5%	18.7%	15.8%	16.9%	20.0%	16.3%	15.4%	18.4%
	PB-LSTM(ID)	17.5%	11.9%	18.3%	13.5%	15.3%	17.9%	15.7%	17.0%	19.0%	16.2%	15.3%	17.9%

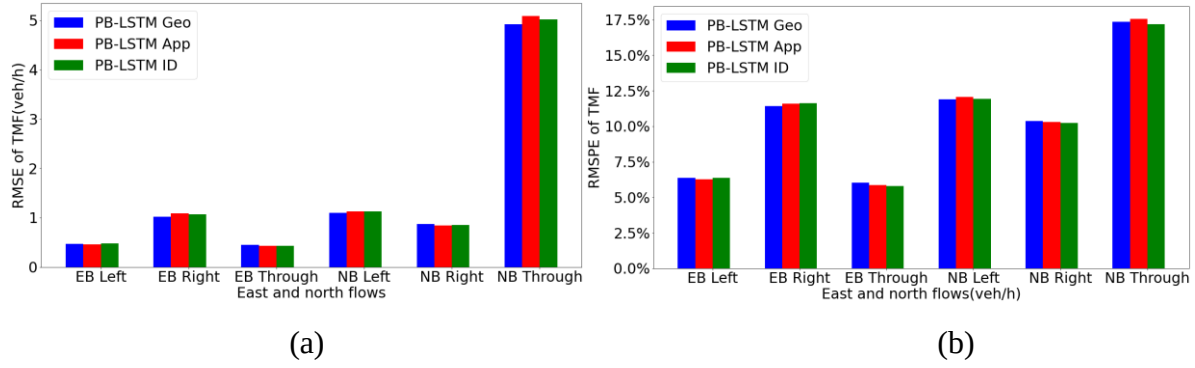


Figure 3.16: Comparison of PB-LSTM with Geometry classification label, PB-LSTM with Approach classification label, PB-LSTM with ID numbers, and normal LSTM models when applied to Main Street and Leisure Centre Driveway

Table 3.5: Model comparison in Intersection Ontario Street South and Childs Drive

		Eastbound			Northbound			Southbound			Westbound		
Turning Movements		Left	Right	Through	Left	Right	Through	Left	Right	Through	Left	Right	Through
Traffic Volume		297751	585761	288633	534409	641840	4575455	277812	320201	4573341	607004	324450	360887
RMSE	PB-LSTM(Geo)	0.474	1.022	0.450	1.100	0.876	4.916	0.971	0.570	5.765	0.748	1.019	0.558
	PB-LSTM(App)	0.462	1.085	0.432	1.132	0.847	5.084	0.941	0.590	5.909	0.715	1.034	0.544
	PB-LSTM(ID)	0.475	1.073	0.427	1.128	0.848	5.012	0.959	0.583	5.836	0.736	0.998	0.560
RMSPE	PB-LSTM(Geo)	6.4%	11.4%	6.0%	11.9%	10.4%	17.4%	11.2%	7.5%	18.5%	9.1%	11.5%	7.1%
	PB-LSTM(App)	6.3%	11.6%	5.9%	12.1%	10.3%	17.6%	11.2%	7.8%	18.8%	9.2%	11.7%	7.0%
	PB-LSTM(ID)	6.4%	11.6%	5.8%	11.9%	10.2%	17.2%	11.2%	7.6%	18.4%	9.1%	11.6%	7.1%

3.4 Conclusions

This research has explored the applications of several state-of-the-art deep learning models for short-term forecasting of intersection turning movement flow (TMFs) based on real-time turning movement counts (TMCs). Specifically, a novel deep neural network model was proposed, which combines the latest LSTM (long short-term memory) model called stacked bidirectional LSTM with the traditional simple MLP, enabling learning of both short and long-term time-varying patterns. The performance of the proposed model was evaluated and compared to several state-of-the-art base models using two years of TMCs at 20 intersections in the city of Milton, Ontario, Canada. A series of experiments were conducted to examine how the performance of the models is affected by several critical model settings including input data range, prediction horizon, and intersections clustering. The main findings and contributions of this research are summarized as follows:

- The proposed PB-LSTM model was shown to have a consistent performance advantage over several state-of-the-art time-series forecasting models, including stacked bidirectional LSTM (SBiLSTM) and MLP.
- Based on a sensitivity analysis of the effect of input data range on model performance, the PB-LSTM model required input data as short as 30 mins to achieve the comparable performance of the SBiLSTM model using a data input range of 60 mins.
- The performance of all models was shown to deteriorate by forecasting horizon; however, the proposed model had a much slower deterioration rate as compared to others.
- To improve the generalization ability of the proposed model, a clustering step was introduced to group intersections of similar attributes and use the grouping as an additional input to the model. Two clustering scenarios were proposed and their effect on the model performance was evaluated. It was found that the model with a consideration of clustering performed better than the one without clustering; however, the performance advantage was moderate at best.

It should be noted that this research has two main limitations. First, the proposed model was found to be still limited in capturing abrupt short-term traffic variations. This could be attributed to either the learning capacity of the model or the inherent random variation of traffic (Chen et al., 2012; Ghanim & Shaaban, 2018). Secondly, this effort did not explore the question of how applicable the proposed model is for forecasting intersection traffic with little training data or those traffic in other cities. Further research is therefore needed to explore other model forms such as graph neural network (Scarselli et al., 2008) and transformers for capturing the network-wide interactions of traffic and examining the predictability issue of traffic in general. The following chapters provide additional efforts in these directions.

Chapter 4: A Deep Learning Model for Intersection-Level Turning Movement Flows Forecasting with Uncertainty Quantification

Chapter 3 presented the development and validation of the Parallel Bidirectional Long Short-Term Memory (PB-LSTM) model, a hybrid neural network framework designed for real-time intersection turning movement flow (TMFs) forecasting. By integrating stacked bidirectional LSTM layers with multilayer perceptron (MLP), the PB-LSTM demonstrated superior performance in capturing complex temporal dependencies and improving forecasting accuracy across diverse intersections and time horizons. However, like most existing traffic forecasting models, the PB-LSTM generates deterministic point predictions, which provide no explicit measure of uncertainty. This limitation poses challenges for traffic management systems that require probabilistic insights to assess risk, optimize adaptive signal control, and mitigate congestion under dynamic, real-world conditions.

To bridge this gap, this chapter shifts focus to uncertainty quantification in TMFs forecasting by introducing DeepAR, a probabilistic deep learning model. Unlike deterministic approaches, DeepAR generates prediction intervals (PIs) by estimating the full probability distribution of future TMFs, enabling explicit quantification of forecasting confidence. The DeepAR is evaluated for its adaptability and reliability in simultaneously processing multiple time series (e.g., cross-sectional traffic movements) and external covariates (e.g., temporal, spatial, and contextual features). Through extensive computational experiments using real-world data from 39 intersections, this study demonstrates how DeepAR addresses the limitations of deterministic models like PB-LSTM, offering actionable insights for traffic operators to balance prediction accuracy with operational risk in Advanced Traffic Management and Information Systems (ATMIS).

The remainder of this chapter is structured as follows. The literature review section provides a comprehensive overview of existing traffic flow forecasting models and successful applications of the DeepAR model. The methodology section details of the DeepAR model used in this study, followed by the case study section, which presents various experiments and sensitivity analyses. The results section discusses the experiments and their findings, and the conclusion section outlines the main findings.

4.1 Literature review

In the past few decades, significant efforts have been made to address the challenge of short-term traffic forecasting, ranging from traditional statistical time-series models to modern machine learning algorithms. The most popular statistical models for traffic forecasting include Autoregressive Integrated Moving Average (ARIMA)(Kumar & Vanajakshi, 2015), Seasonal ARIMA(Williams & Hoel, 2003), and Prophet(ChikkaKrishna et al., 2022), which were found to be reliable for dealing with relatively stationary traffic but ineffective for handling highly varied, non-stationary traffic with complex seasonal trends and non-recurrent variations. With the rise of machine learning techniques, models such as Support Vector Regression (SVR)(Wei & Liu, 2013), K-Nearest Neighbors (K-NN)(Sun et al., 2018), and Random Forest(Liu & Wu, 2017) have demonstrated superior performance in traffic forecasting compared to traditional statistical models. However, these models have limitations in handling big data problems and combining data from different sources effectively. Deep learning models have emerged as the dominant technique for traffic forecasting due to their ability to handle large and complex datasets. Models like Multilayer Perception (MLP)(Qin et al., 2023), Convolutional Neural Networks (CNNs)(Zhang et al., 2019), Recurrent Neural Networks (RNNs)(Cui et al., 2019), and Long Short-Term Memory (LSTM)(Zhao et al., 2017) have shown promising results in various forecasting problems. Additionally, state-of-the-art deep learning models such as Transformer (Xu et al., 2020), Graph Neural Networks (GNNs)(Jiang & Luo, 2022), and Spatiotemporal Graph Convolutional Networks (ST-GCNs)(Yu et al., 2017) have also been applied for traffic flow forecasting.

However, there is a notable gap in the literature concerning the problem of intersection Turning Movement Flows (TMFs) forecasting. Most existing studies(Karnati et al., 2021; Z. Li et al., 2019) use simulation data from SUMO and VISSIM for training their models. Karnati et al.(Karnati et al., 2021)developed a three layers MLP model using Turning Movement Counts (TMCs) data from a SUMO simulation model. Similarly, Li et al.(Li, Ban, et al., 2020)proposed a novel Deep Intersection Spatial-Temporal Network (DISTN) combining LSTM and CNNs models for TMFs forecasting based on VISSIM simulation data. The results of these studies demonstrate the high-performance advantage of deep learning models for TMFs forecasting; however, their findings have not been validated using real-world data. Moreover, some studies(Ghanim & Shaaban, 2018) have attempted to build deep learning models based on real-world data, but the lack of evaluation on hold-out data raises questions about the performance of these models on new locations. Mahamoud et al. (Mahmoud et al., 2022) proposed an

ensemble learning model to estimate the next cycle traffic turning movements in traffic signal cycle level, using real-time collected turning movement counts data and signal timing from 19 intersections along 2 corridors in Florida, USA. Their results showed that the ensemble model outperformed the linear regression baseline models with MAPE 9.53% and 4.6% for through and left-turn movements, respectively. The single next cycle level forecasting can be use for dynamic traffic signal control, but their study failed to evaluate their model using hold-out data; as a result, it is unclear how their model would perform when applied to a new intersection. Furthermore, none of the past studies have attempted to quantify uncertainty in forecasting, leaving a significant gap in understanding the reliability of a forecasting model.

To address these limitations, this study applies the DeepAR (Salinas et al., 2020) for TMFs forecasting and conducts a comprehensive forecasting uncertainty analysis. The DeepAR model, a state-of-the-art time series forecasting model developed by Amazon, has shown successful applications in various domains such as demand forecasting (Khan et al., 2020), stock market prediction (Consoli et al., 2022), energy consumption forecasting (Chakrabarti et al., 2022), weather forecasting (Liao & Liang, 2021) and railway freight volume forecasting (Zhao et al., 2023). It can provide accurate forecasting not only based on observed values through an autoregressive encoder-decoder mechanism (Gregor et al., 2014), but also learn cross-sectional features from different groups of time series and incorporate covariates as additional time series. Additionally, DeepAR providing not only future time series estimates but also their probability distributions. This allows practitioners to compute quantile estimates and effectively quantify the uncertainty associated with the forecasting.

4.2 Problem description and DeepAR model

Consider a set of n_s intersections in an urban road network, denoted as $\mathcal{S}$, with each intersection, represented by s , where $s \in \mathcal{S}$, having n_m movements. For a four-leg intersection, as depicted in Figure 4.1, there are a total of 12 movements. This encompasses three types of movements: right turn (R), left turn (L), and straight through (T). These movements apply to four different directions or bounds: eastbound (EB), westbound (WB), northbound (NB), and southbound (SB).

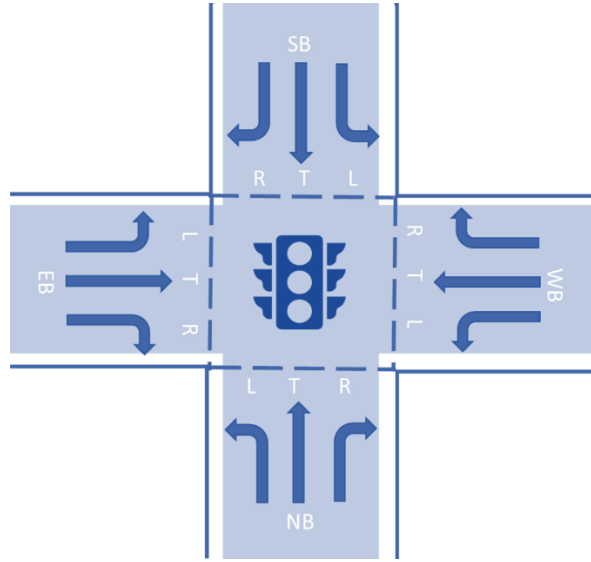


Figure 4.1: Traffic flow from different directions and turning movements

Assume that all these intersections are instrumented with continuous traffic count data, aggregated at a fixed time interval (e.g., 5-minute), on all turning movement flows (TMFs). The TMFs therefore consist of $n_s \cdot n_m$ time series values and a total of n_t past intervals are considered, representing the observed data for the n_m turning movements at n_s intersections over the past n_t time intervals. Let $\mathbf{X}_{s,(t_0,t)}$ be the TMFs time series for a specific intersection s over a specific time period from t_0 to t_0, t (Equation (4.1)).

$$\mathbf{X}_{s,(t_0,t)} = \begin{pmatrix} x_{t_0,1} & x_{t_1,1} & \dots & x_{t,t,1} \\ x_{t_0,2} & x_{t_1,2} & \dots & x_{t,t,2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{t_0,m} & x_{t_1,m} & \dots & x_{t,t,m} \end{pmatrix} \quad (4.1)$$

Where $x_{t,m}$ is the record of turning movement m at time t at intersection s (note that the subscript s is dropped for convenience).

The TMFs forecasting problem can then be defined as using n_h intervals (input data range) of TMFs data from time interval $t - h$ to the time interval t to predict the future n_p intervals (forecasting horizon) of TMFs from interval $t + 1$ to $t + p$ of TMFs by TMFs forecasting model $F(\cdot)$, as shown in Equation (4.2).

$$X_{s,(t+1,t+p)} = F(X_{s,(t-h,t)}) = F \begin{pmatrix} x_{t-h,1} & x_{t-h+1,1} & \dots & x_{t,1} \\ x_{t-h,2} & x_{t-h+1,2} & \dots & x_{t,2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{t-h,m} & x_{t-h+1,m} & \dots & x_{t,m} \end{pmatrix} \quad (4.2)$$

In addition to the TMFs data, the TMFs pattern can also be influenced by spatial features and other covariates. Therefore, it is necessary to include spatial features $V_{spatial}$ and covariates $V_{cov,t}$ as additional input data. Spatial features $V_{spatial}$ could includes the location of the intersections which are not varied by time, while $V_{cov,t}$ represents the features varied by the time, such as, weather, holidays, and special events. Figure 4.2 illustrates the key aspects of the forecasting problem of a single time series data. In order to predict the next n_p intervals of TMFs data $X_{s,(t+1,t+p)}$, the forecasting model takes past TMFs data $X_{s,(t-h,t)}$, spatial features $V_{spatial}$, and covariates $V_{cov,(t-h,t)}$ as input. The inputs could also include future known covariates $V_{cov,(t+1,t+p)}$ as well as spatial features $V_{spatial}$, which remain constant over time.

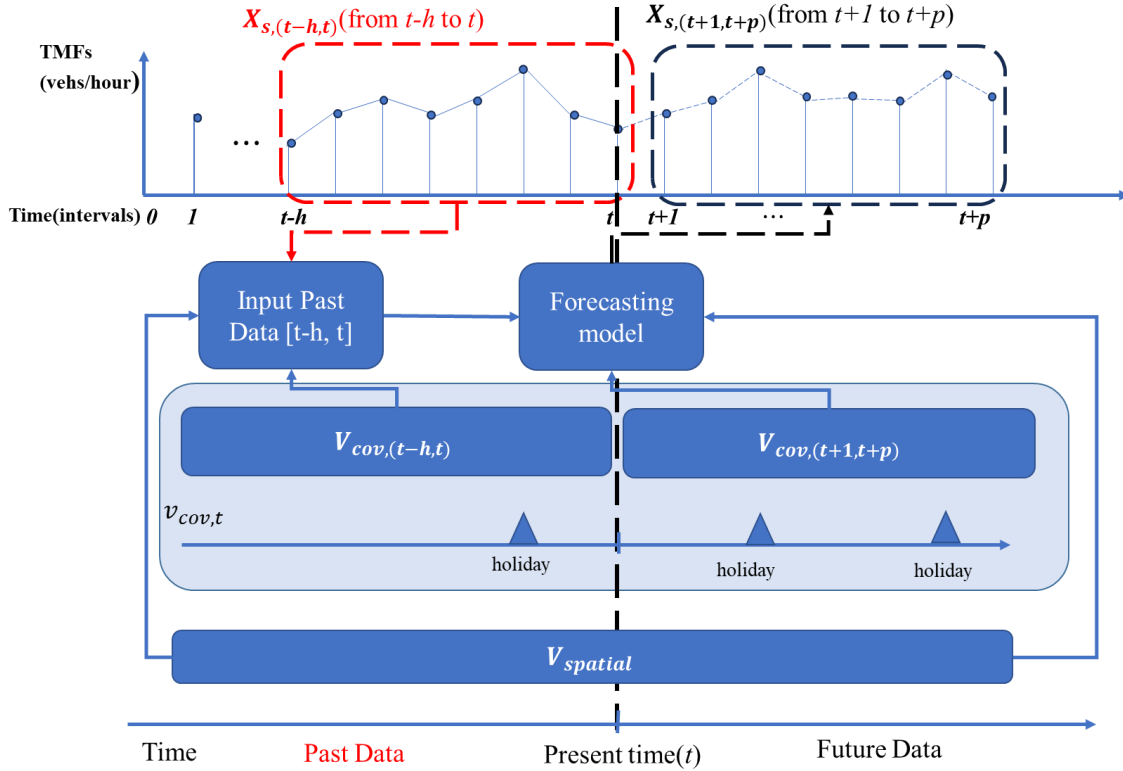


Figure 4.2: The TMFs forecasting problem

The ultimate goal of this research is to develop an accurate and reliable mapping function or deep learning model that can be applied for short-term traffic flow forecasting at an intersection movement level.

In this research, we leverage DeepAR, a deep learning model utilizes Long Short-Term Memory (LSTM) cells within its encoder-decoder architecture. DeepAR offers three key advantages over traditional forecasting models: (1) Cross-Sectional Learning: DeepAR can capture relationships and dependencies between different turning movement time series data (e.g., how northbound go through traffic flow might influence westbound turning right flow). This cross-sectional learning capability allows the model to make more informed predictions. (2) Incorporation of External Covariates: Beyond historical TMFS data, DeepAR can integrate external covariates (e.g., locations, holidays) as additional time series inputs. This allows the model to account for external factors that might influence traffic patterns. (3) Probabilistic Forecasting: Unlike traditional models that provide single point forecasts, DeepAR offers probabilistic forecasting by estimating the entire probability distribution parameters for the next time step. This means it generates not just a single predicted value but also estimates the uncertainty associated with that prediction. This information about prediction confidence is valuable for decision-making purposes. Figure 4.3 illustrates the fundamental working mechanism of the DeepAR architecture for both training (left) and prediction (right).

During training, at each time interval t , the network receives input (lower red rectangle in Figure 4.3) including observed external covariates $\mathbf{v}_{cov,t}$, the traffic flow of turning movement m at the previous time interval $\mathbf{x}_{t-1,m}$, and the previous network hidden state $\mathbf{h}_{t-1,m}$. For each turning movement series, the encoder network utilise LSTM (upper red rectangle in Figure 4.3) produces the new hidden state $\mathbf{h}_{t,m} = \mathbf{h}(\mathbf{h}_{t-1,m}, \mathbf{x}_{t-1,m}, \mathbf{v}_{cov,t}, \theta)$, where θ represents the model parameters. The hidden state $\mathbf{h}_{t,m}$ is then used to compute the parameter $\theta_{t,m} = \theta(\mathbf{h}_{t,m}, \theta)$ of the likelihood function $\ell(\mathbf{x}_{t,m}|\theta_{t,m})$, which is used for training the θ . The model parameter θ will be learned from by maximizing the log-likelihood of $\theta_{t,m}$ over all the turning movements and the entire time range to leverage the cross-learning effects.

During the forecasting process, DeepAR utilizes a rolling horizon strategy. When we are going to make the prediction after time interval t , the observed time series history $\mathbf{x}_{t-1,m}$ is fed into the model as historical input data with $\mathbf{v}_{cov,t}$ at time interval t . Then the DeepAR model estimates a future probability

distribution $\ell(\cdot | \theta_{t,m})$ at interval t and draw a prediction $\hat{x}_{t,m}$ sampled from $\hat{x}_{t,m} \sim \ell(\cdot | \theta_{t,m})$. The $\hat{x}_{t,m}$ will be feed back as input for the next time interval $t + 1$, continuing until the end of the prediction. This process generates a single sample trace. By generating multiple sample traces, DeepAR can capture the uncertainty inherent in the predictions and provide a probabilistic forecast. The distribution of the predicted values allows for estimating confidence intervals and assessing the range of possible outcomes.

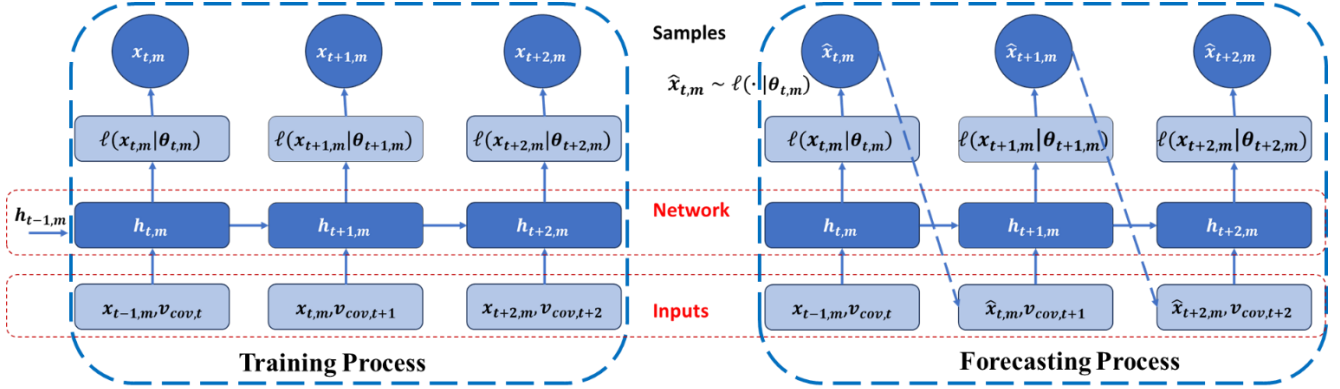


Figure 4.3: DeepAR model architecture(Salinas et al., 2020)

DeepAR models are characterized by their ability to provide probabilistic forecasting, capture long-term dependencies and cross-sectional features, make multiple horizon forecasting, and handle multiple covariates in data (Salinas et al., 2020). The proposed research will explore the application of DeepAR models for TMFs forecasting with a specific focus on their features such as probabilistic forecasting, handling of seasonality and trends, and integration of time-series covariates.

4.3 Computational experiments and results

The DeepAR models were implemented using a PyTorch backend (Paszke et al., 2019). To ensure model robustness, cross-validation was applied at each iteration, with 20% of the training dataset randomly selected for model validation. The DeepAR model was configured with 2 layers of Long Short Term Memory (LSTM) units, and the Adam optimizer (Edwards et al., 2008) was employed for model training. The model's learning rate was determined using the built-in learning rate finder from PyTorch Lightning (Falcon, 2019).

Based on the preliminary results that compared different historical data ranges as input, it was found that utilizing a 4-hour (48 intervals) historical input data range proved to be achieved the best results in terms of accuracy. The average training time for forecasting the next 5 minutes (1 interval) was approximately 8 seconds per iteration per training batch (batch size = 1024 records) on a NVIDIA GeForce V100. During testing on the test dataset, the forecasting process was completed within 6103 seconds, implying that each prediction was performed in under 6.9×10^{-4} seconds. The subsequent section presents the key findings from various computational experiments.

4.3.1 Case study data description

In this research, 15 months real-time intersection traffic turning flow data from March 1st, 2022, to May 31st, 2023, were collected from 39 intersections in City of Milton, a midsize town with 132,979 population in the Great Toronto Area (GTA) in Ontario, Canada. The data were obtained by Miovision SmartSense systems, which is a video-based detection system that provides continuous turning movement counts. Data is made available through an API that aggregates count of individual vehicles on a minute-by-minute basis.

Figure 4.4 shows the location of City of Milton which is connected to City of Toronto by Highway 401, and the study area, featuring 34 intersections that were utilized to train the DeepAR model, which includes a total of 34×12 TMFs time series. An additional 5 intersections were set aside for testing purposes as hold-out test intersections. To process the data for model training and testing, the initial Turning Movement Count (TMC) data, were further aggregated into 5-minute intervals. Subsequently, we converted this aggregated data into Turning Movement Flows (TMFs) data, expressed in vehicles per interval (vehs/interval). The resulting TMFs dataset for the 34 intersections was divided into a training dataset, which encompasses 365 days of traffic data (3,574,080 records), and a separate test dataset, covering 90 days of traffic data (881,280 records).

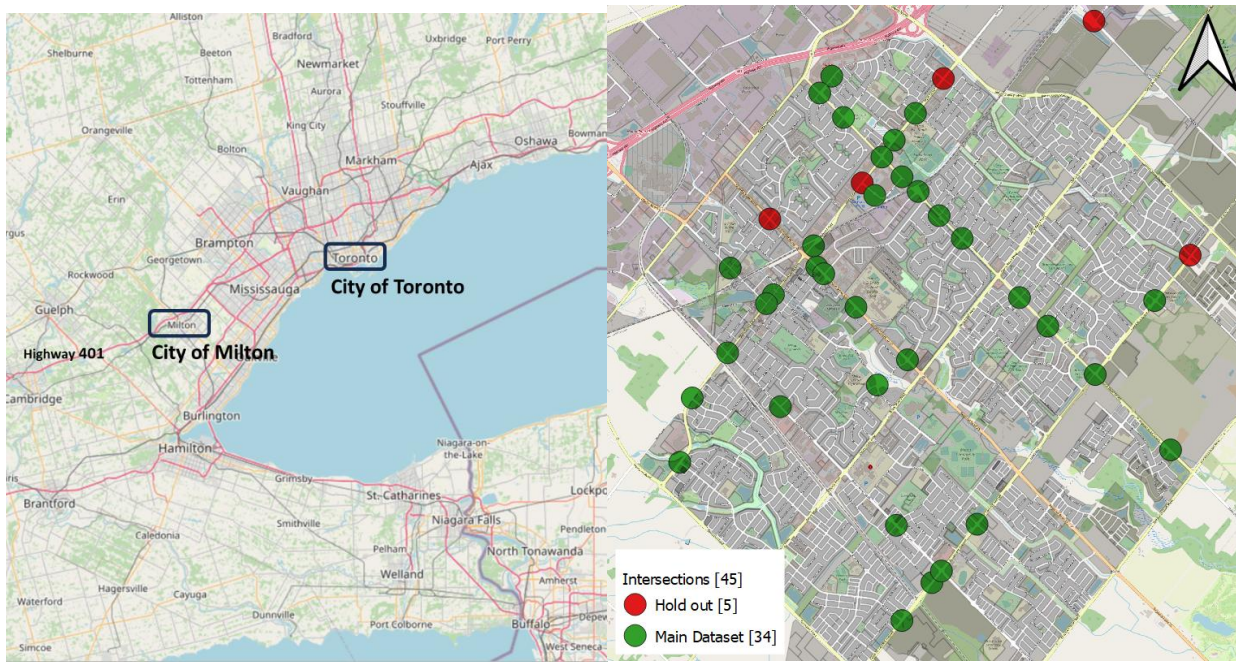


Figure 4.4: The study area (left: location of City of Milton; right: the locations of collected intersections)

The inputs to the DeepAR model encompass not only the historical TMFs time series data but also include $V_{spatial}$ and $V_{cov,t}$ as additional time series as covariates, which are summarized in Table 4.1.

Table 4.1: Summary of input variables

Variable Type	Variable	Description
$V_{spatial}$	Location	Intersection longitude and latitude
	Street	Intersected street names (Name 1, Name 2)
$V_{cov,t}$	Holiday	Public holidays in Ontario
	Hour	The hour of day
	Weekday	The day of week
	Month	The month of year
$X_{s,(t_0,t)}$	TMFs	Turning movement time series data

4.3.2 Performance metrics

To assess the performance of DeepAR Root Mean Square Error (RMSE) and Root Mean Squared Percentage Error (RMSPE) were used as metrics to evaluate the forecasting accuracy. Additionally, reliability and sharpness measures (Lin et al., 2018) were employed to evaluate the forecasting

uncertainty by calculating the 95% forecasting confidence interval for predictions and comparing them to the actual TMFs.

For the reliability evaluation, we considered the 95% Prediction Interval (PI) that provides a lower bound $\hat{L}_i$ and an upper bound $\hat{U}_i$ for the forecasted future target value $\hat{X}_i$, the probability that the observed future targets X_i all within the PI should be equal to 95%. However, in practice, the actual Prediction Interval Coverage Probability (PICP) may deviate from the nominal prediction interval confidence level (95% in this research). The reliability metric aims to quantify the difference between PICP and the nominal prediction interval confidence level, serving as a measure of how well the forecasting uncertainty aligns with the intended confidence level:

$$PICP = \frac{1}{N} \sum_i^N D_i \quad (4.3)$$

$$R^\alpha = abs(PICP - 95\%) \quad (4.4)$$

Where:

N represents the size of the evaluation dataset, and N is calculated as the product of n_s , n_t , and n_m ; D_i is a dummy variable that takes the value 1 if the observed future target is within the PI, and 0 otherwise.; R^α denotes the absolute coverage error used for the reliability evaluation criterion, where a smaller value of R^α indicates higher reliability.

Reliability solely focuses on the coverage probability, and a model with a wider PI will naturally exhibit higher reliability. However, a wide PI may not necessarily offer accurate quantification of uncertainties inherent in real-world processes. A robust forecasting model should not only provide reliable confidence intervals but also ensure they are sharp.

To quantify the width of the PI, denoted as WI_i . we measure the distance between the upper bound and the lower bound using Equation (4.5):

$$WI_i = \hat{U}_i - \hat{L}_i \quad (4.5)$$

The sharpness of the PI can be denoted by

$$S_i = \begin{cases} w_1 WI_i + w_2 [\hat{L}_i - X_i], & \text{if } X_i < \hat{L}_i \\ w_1 WI_i, & \text{if } X_i \in [\hat{L}_i, \hat{U}_i] \\ w_1 WI_i + w_2 [X_i - \hat{U}_i], & \text{if } X_i > \hat{U}_i \end{cases} \quad (4.6)$$

Where:

w_1 and w_2 are two user-defined weights.

Equation (4.6) considers the width of the PI, WI_i , which is weighted by w_1 for all three different scenarios. Additionally, to prevent any bias errors caused by an overly narrow PI, an extra penalty is calculated when the observed future targets fall outside the PI's bounds. This penalty is determined by the distance of that point to the bound and is represented by w_2 . In this research, is included to prevent the bias error by PI becoming too “narrow”. In this research, we set the $w_1 = 1$ and $w_2 = 2$ based on the existing study(Lin et al., 2018).

The sharpness of the PI across the entire dataset can be evaluated by computing the average of the normalized S_i , represented by $S_{i,norm}$ using Equations (4.7) and Equation (4.8):

$$S = \frac{1}{N} \sum_i^N S_{i,norm} \quad (4.7)$$

$$S_{i,norm} = \frac{S_i - \min(S_i)}{\max(S_i) - \min(S_i)} \quad (4.8)$$

4.3.3 Model performance

We first trained the DeepAR model on a dataset comprising 34 intersections for one-interval (5 mins) forecasting and compared with a baseline model ARIMA(Kumar & Vanajakshi, 2015) and four other forecasting models including Transformer(Lim et al., 2021), Stacked Bidirectional LSTM (SBiLSTM)(Schuster & Paliwal, 1997), Parallel Stacked Bidirectional LSTM (PB-LSTM)(C. Zhang et al., 2023), Multilayer Perceptron (MLP)(LeCun et al., 2015), the results on test dataset are in Table 4.2. The results indicates that both the Transformer and DeepAR models significantly outperform the

baseline ARIMA model and other models. Specifically, DeepAR demonstrates a low RMSE (1.7753), comparable to the Transformer's RMSE (1.6879), but it surpasses the Transformer in terms of RMSPE, with a remarkably smaller percentage error of 9.6269% compared to the Transformer's 14.1793%. This suggests that DeepAR is not only effective in providing accurate point forecasts but also excels in capturing extreme values, which is crucial for practical traffic management applications.

Figure 4.5 displays a one-day forecasting of all 12 turning movements (same as Figure 4.1 shown) intersection flow using rolling horizon prediction for the eastbound of an example of 34 intersections, Main Street East and Maple/Sinclair Drive, on 17th August 2022. It is worth noting that the example intersection is situated between a residential area (to the East) and a local business area (to the West). This unique location leads to a morning peak hour in the westbound direction and an evening peak hour in the eastbound direction.

Table 4.2: Model comparison

	ARIMA	DeepAR	Transformer	SBiLSTM	PB-LSTM	MLP
RMSE (vehs/interval)	3.4034	1.7753	1.6879	2.6440	2.2489	7.9743
RMSPE (%)	38.2192	9.6269	14.1793	15.2827	11.5529	48.4623

The Figure 4.5 illustrates the predicted flows in comparison to the observed flow, along with the 95% forecasting PI. The results demonstrate that the DeepAR model effectively captures the trend of the observed data and provides a reliable PI. However, DeepAR faces challenges in accurately predicting the rapid increase in traffic flow during peak hours with point forecasting alone. Nonetheless, by incorporating forecasting confidence intervals, DeepAR can capture these extreme fluctuations, assisting traffic management in avoiding underestimating peak-hour turning flows. Additionally, we observed that the DeepAR produces sharper PI when dealing with intersections experiencing larger average traffic volumes, as opposed to predictions for different movements.

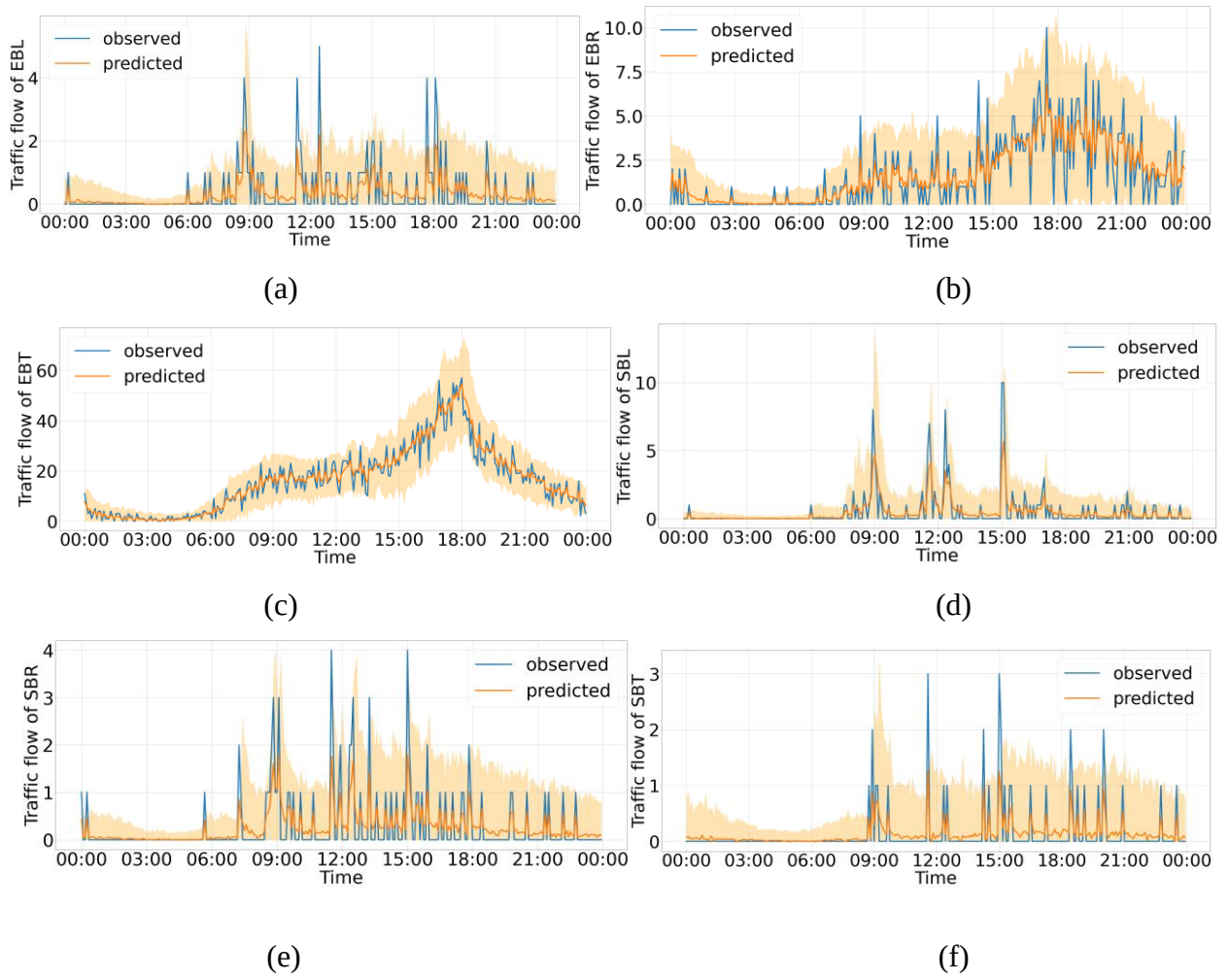


Figure 4.5: Sample of observed and predicted TMFs (veh/ interval)

4.3.4 Comparison with Transformer and ARIMA for multistep forecasting

As the DeepAR model also provides probabilistic forecasting, in this section we conduct a comprehensive comparison between the DeepAR model, the state-of-the-art model, Transformer, and the baseline autoregressive statistical model, ARIMA, to assess its forecasting accuracy and uncertainty in multistep predictions. We evaluate both models using key performance metrics, including RMSE, RMSPE, Reliability (R^a), and Sharpness. The comparison is performed on a rolling horizon forecasting approach, covering various forecasting horizons (one-interval, two- intervals, four- intervals, eight-intervals, twelve- intervals, thirty-six- intervals, and one hundred twenty- intervals), where each interval corresponds to a five-minute period.

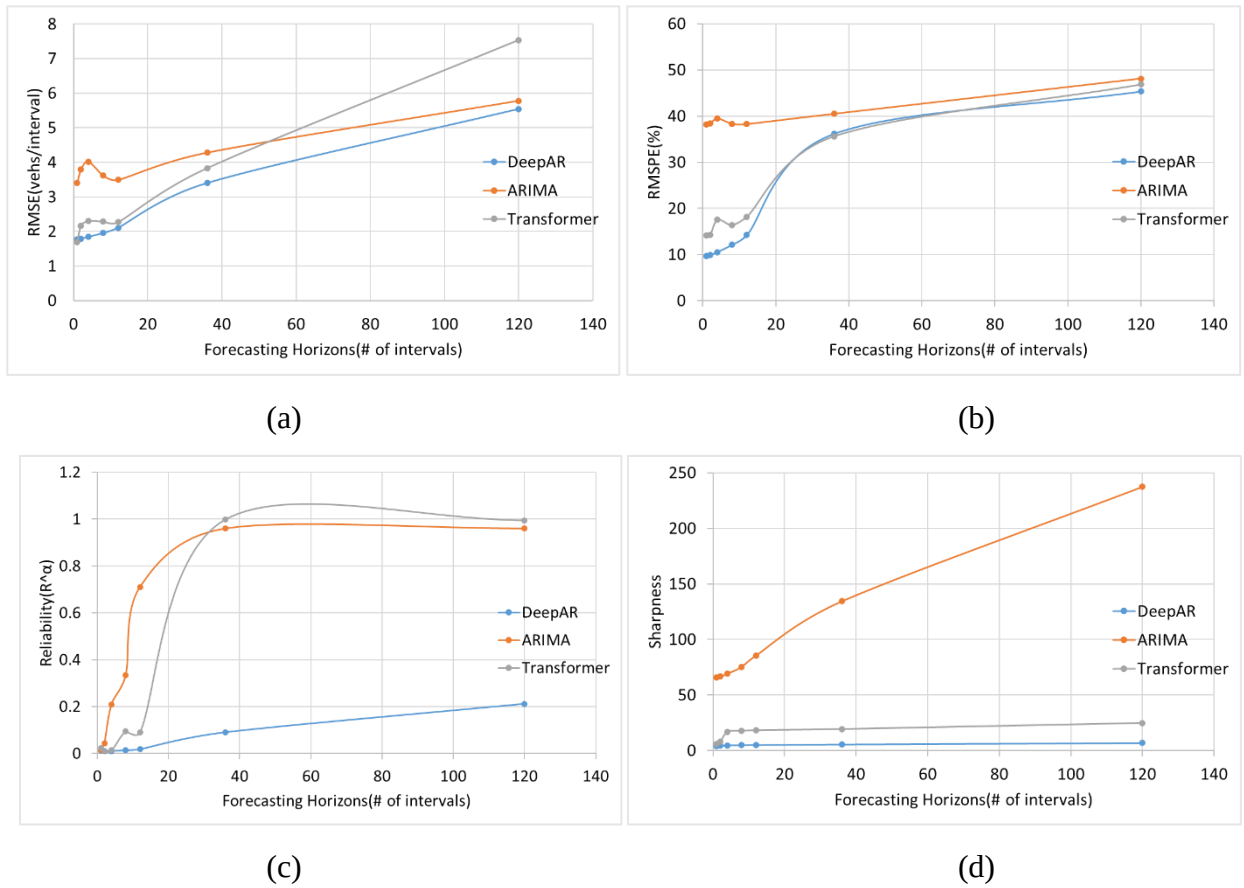


Figure 4.6: Forecasting performance comparison between DeepAR, Transformer, and ARIMA with different forecasting horizons

Figure 4.6 shows the comparison results, and we can find that the DeepAR model outperforms ARIMA and Transformer across multiple metrics and forecasting intervals. DeepAR consistently achieves lower RMSE and RMSPE values, demonstrating superior accuracy and a better ability to handle extreme values. Specifically, DeepAR's RMSE ranges from 1.7753 to 5.5466, and its RMSPE ranges from 9.6269% to 45.3601%, significantly outperforming ARIMA, which has RMSE values from 3.4034 to 5.7794 and RMSPE values from 38.2192% to 48.1321%. The Transformer model, while competitive with the lowest RMSE for shorter intervals, shows higher RMSPE and less reliability in longer-term forecasting compared to DeepAR. Additionally, DeepAR exhibits lower reliability values (R^α) ranging from 0.0126 to 0.2123 and smaller sharpness values (4.2831 to 6.8821), indicating more accurate and tighter prediction intervals. In contrast, ARIMA's high sharpness values (66.1145 to 237.6057) reflect much wider intervals and less precision. The performance degradation over longer forecasting horizons is more pronounced in ARIMA, whereas DeepAR and Transformer maintain better accuracy and uncertainty estimation. Overall, DeepAR's superior performance across these metrics makes it the most

effective model for traffic forecasting, providing more robust and reliable predictions than ARIMA and Transformer.

4.3.5 Effect of time varying covariates

The DeepAR model can incorporate covariates, such as hours of the day, days of the week, and months of the year, to capture long-term seasonality and trends in turning movement flow (TMFs) forecasting. Including these time covariates may enhance the model's accuracy and reduces uncertainty in predictions. In this experiment, we constructed two DeepAR models for one-interval (5 mins) forecasting: the first model utilised covariates $V_{cov,in}$ showed in Table 4.2 as additional time series during training, while the second model was trained without these covariates. The result is presented in Figure 4.7.

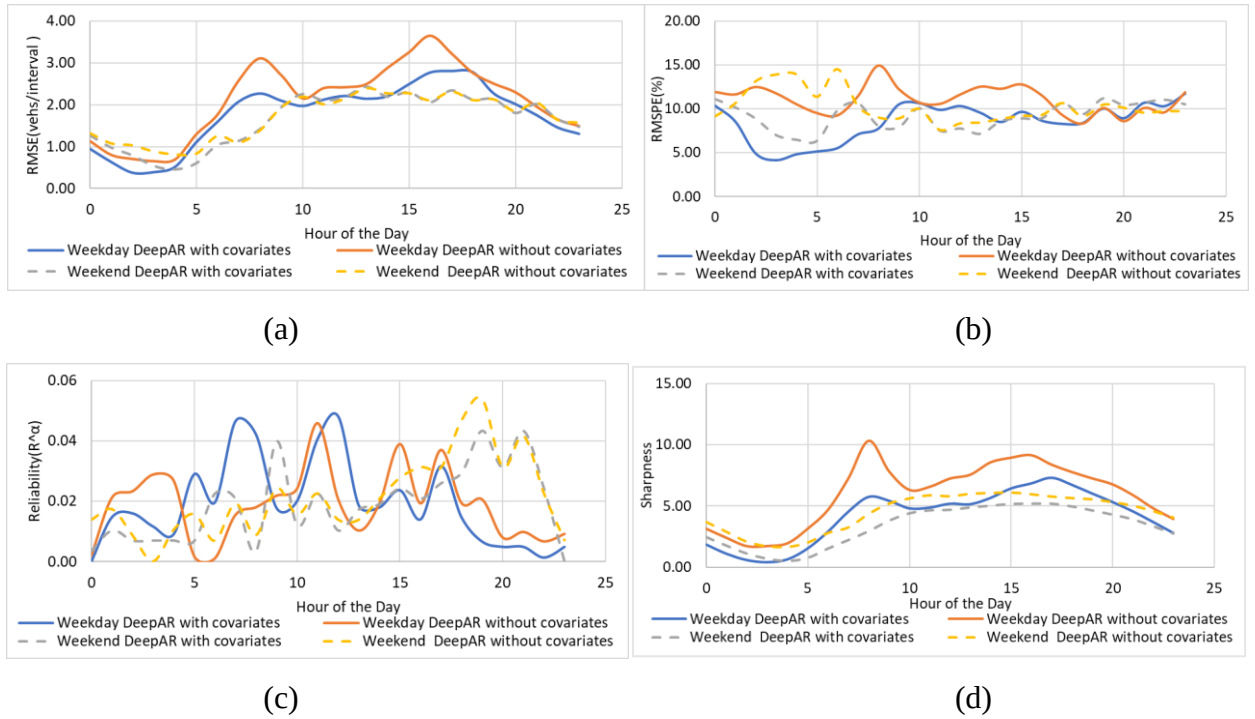


Figure 4.7: Forecasting performance for DeepAR model with covariates and without covariates

During the weekdays, we can find the DeepAR model trained with covariates consistently has lower RMSE and RMSPE values, which means it outperforms the model trained without covariates. This indicates that incorporating additional time series covariates (day of the week, hour of the day, and month of the year) leads to improved point prediction accuracy and reduced percentage error in the

forecasting. Similar as RMSE and RMSPE, the sharpness of the prediction intervals for the model trained with covariates is lower, which indicates that the covariate model produces tighter prediction intervals, implying a lower level of uncertainty in the forecasting. The superiority is most prominent during peak hours when traffic congestion and complex patterns are more challenging to predict accurately. The covariates effectively capture the temporal patterns during weekdays, contributing to improved performance. However, RMSPE and Reliability values fluctuate across different hours of the day. It's possible that certain hours have more predictable patterns, leading to lower RMSPE and more reliable prediction intervals, while other hours might have more variability, resulting in higher RMSPE and less reliable forecasting. On weekends, the difference between the two DeepAR models is less pronounced compared to weekdays. This observation aligns with the notion that weekends typically have simpler traffic patterns compared to weekdays, as there are no distinct peak hours. The absence of peak hours makes the weekend scenarios more straightforward to forecast, leading to less disparity between the DeepAR models with and without covariates. Furthermore, most of the data is gathered from intersections located near central business districts or downtown areas, as mandated by municipal requirements. These specific areas tend to experience a notable decline in traffic volume during weekends, leading to less predictable traffic patterns. Consequently, the influence of covariates during these periods is likely to be diminished.

In summary, incorporating covariates into the DeepAR model provides valuable insights into temporal patterns and enhances prediction accuracy during peak hours and weekdays. The fluctuations in RMSPE and Reliability across different hours indicate the varying complexity of traffic patterns throughout the day. On weekends, where traffic patterns are relatively simpler, the performance gap between the models with and without covariates is less significant. Ultimately, using covariates in the DeepAR model proves to be a valuable approach for improving traffic forecasting accuracy, especially in complex scenarios characterized by peak hours and weekdays.

4.3.6 Effect of cross-sectional features

The architecture of DeepAR incorporates a shared representation layer that processes information from multiple time series simultaneously. This shared representation enables the model to extract common features and trends across different groups of time series data, effectively capturing cross-sectional features present in the data.

To explore the impact of using multiple turning movement groups in training, we compared a DeepAR model trained by multiple turning movement groups (each turning movement treated as a time series group) with a group of DeepAR models trained by single turning movements (each turning movement trained as a separate DeepAR model). In comparison, we calculated the RMSE, RMSPE, Reliability (R^α) and Sharpness of one-interval (5 mins) forecasting, and the result was shown in Figure 4.8.

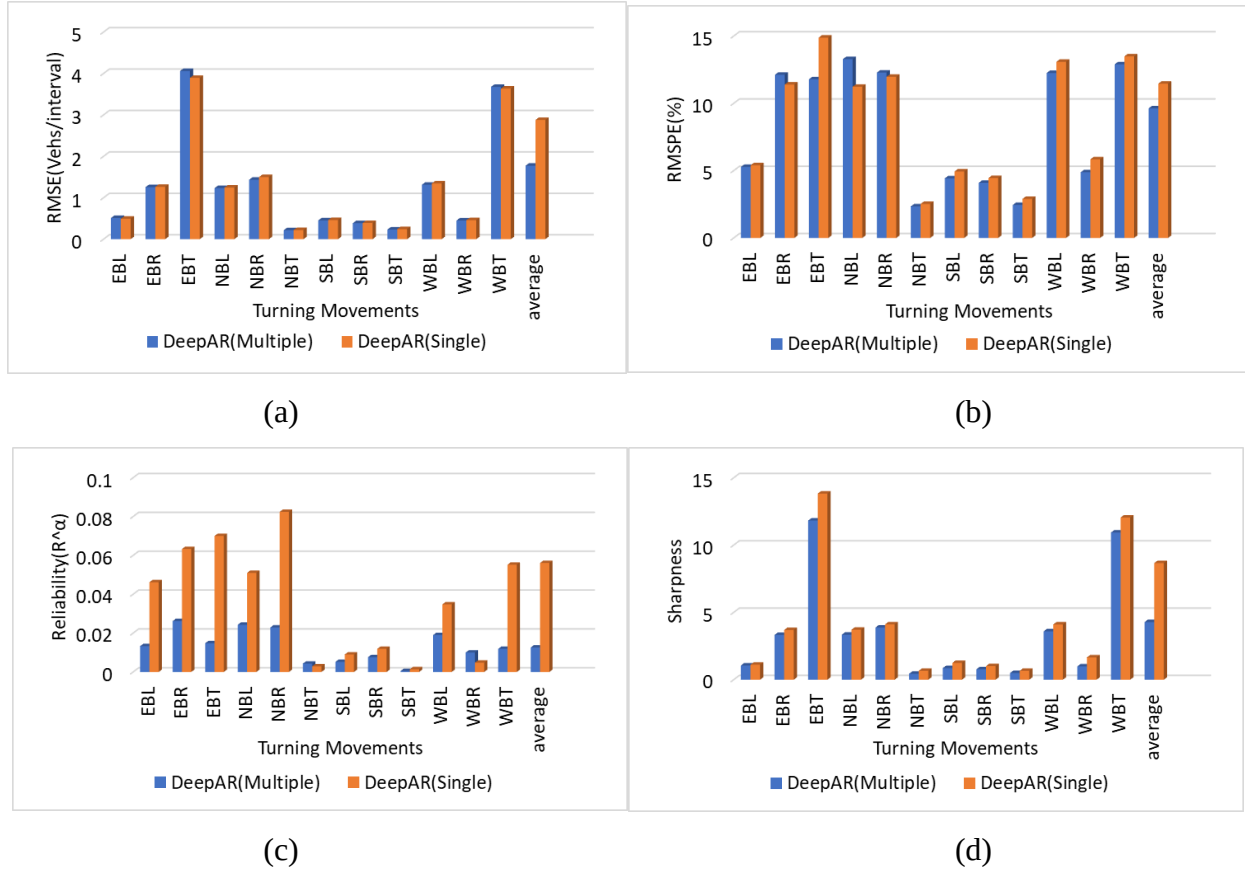


Figure 4.8: Forecasting performance for DeepAR trained by multiple movements and by single movement

From the Figure 4.8, we observed that the RMSE and RMSPE values for each turning movement exhibit similarities when comparing DeepAR trained on all movement groups (multiple turning movements) and DeepAR trained on single movement groups (each turning movement separately). However, in many cases, DeepAR trained on all movement groups displays slightly lower average RMSE and RMSPE values. This discrepancy can be attributed to the inherent variability in scale and size among the different turning movements in the dataset. The Reliability (R^α) values of the DeepAR model trained on all movement groups are consistently lower than the model trained on single movement groups in both

average and each turning movement. This indicates that the prediction intervals of the DeepAR model trained on all movement groups are better calibrated and provide more accurate uncertainty estimation compared to the model trained on single movement groups. Additionally, the Sharpness values for the DeepAR model trained on all movement groups tend to be lower than those of the model trained on single movement groups, which is more significant for Eastbound Through (EBT) and Westbound Through (WBT) movements. This suggests that the prediction intervals of the DeepAR model trained on all movement groups are slightly tighter and more informative compared to the model trained on single movement groups.

By comparing the average RMSE, RMSPE, Reliability (R^α), and Sharpness, the results suggest that using cross-sectional features from multiple turning movement groups in the DeepAR model leads to limited improved forecasting performance in terms of accuracy (RMSE, RMSPE), but major better reliability, and sharpness. The ability of DeepAR to learn cross-sectional features from different groups of time series data enhances its forecasting capabilities and provides more reliable predictions compared to using separate models for each turning movement.

4.3.7 Transferability analysis with hold-out intersections

The DeepAR model has demonstrated exceptional transferability and reliable performance, particularly in cold start scenarios (Liao & Liang, 2021; Salinas et al., 2020; Zhao et al., 2023). In this experiment, we used the 5 hold-out intersections to fine tune the trained DeepAR model for one-interval (5 mins) forecasting. The DeepAR model was trained by 34 intersections used before, and the 5 hold-out intersections are divided into training dataset (524,160 records) and test dataset (131,040 records) also by 80% and 20%, respectively. We have considered 5 scenarios by using different proportion training dataset for fine tuning: cold start (0 record), 20 % training (104,832 records), 40 % training (209,664 records), 60 % training (314,496 records), 80 % training (419,328 records), and 100 % training (524,160 records). The impact of the training dataset proportion on hold-out intersection forecasting is summarized in Figure 4.9, and a comparison with the forecasting performance on other intersections is presented in Table 4.3.

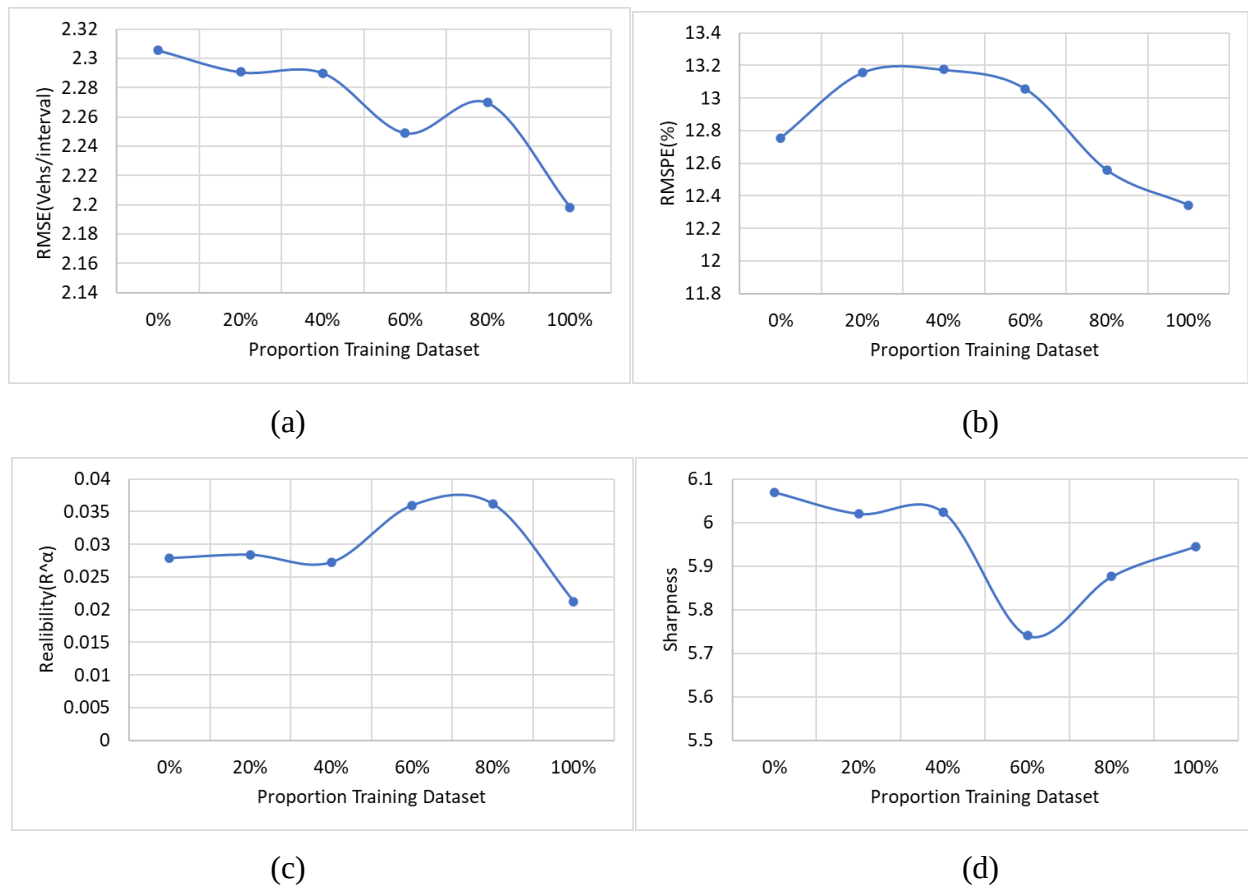


Figure 4.9: Forecasting performance for DeepAR model on hold-out intersections

Table 4.3: Forecasting performance for DeepAR model on hold-out intersections

		RMSE	RMSPE	Reliability	Sharpness
		(vehs/ interval)	(%)	(R^α)	
Hold-out	0% Training	2.3056	12.7567	0.0279	6.0692
	20% Training	2.2907	13.1575	0.0284	6.0199
	40% Training	2.2901	13.1748	0.0273	6.0236
	60% Training	2.2490	13.0565	0.0360	5.7400
	80% Training	2.2697	12.5559	0.0362	5.8761
	100% Training	2.1984	12.3416	0.0213	5.9453
Training Intersection		1.7753	9.6269	0.0126	4.2831

Upon examining the data, it is evident that the RMSE and RMSPE values decrease as the proportion of training data for fine-tuning increases from 0% to 100%. This indicates that increasing the training data helps improve the accuracy of the DeepAR model's point predictions and reduces the percentage error in the forecasting. The Reliability (R^α), values fluctuate as the proportion of training data increases. The

values slightly decrease from 0% to 40% training data, then increase for 60% and 80% training data, and finally decrease again for 100% training data. This shows that the reliability of the prediction intervals is not strictly tied to the proportion of training data. Other factors, such as the characteristics of the hold-out intersections and the fine-tuning process, might influence the reliability. The sharpness of the prediction intervals exhibits a non-monotonic trend. It decreases as the proportion of training data increases from 0% to 60%, indicating that tighter prediction intervals are obtained with more training data. However, the sharpness increases slightly from 80% training data to 100% training data. This suggests that overly tight or overly wide prediction intervals may result from using too much or too little training data. Furthermore, when comparing the cold start scenario with the forecasting performance on the other 34 training intersections trained at the beginning, we observe only a slight decrease in accuracy and uncertainty. This finding indicates that the performance of the DeepAR model demonstrates satisfactory transferability and can be successfully applied to new intersections in same municipality without fine tuning.

In summary, while the RMSE and RMSPE values show a consistent decreasing trend with increasing training data, the trends for reliability and sharpness are not as straightforward. The fluctuations in reliability and sharpness indicate that the choice of the proportion of training data for fine-tuning should be carefully considered and might depend on the specific characteristics of the hold-out intersection datasets.

4.4 Conclusions

This study explored the application of the DeepAR approach for real-time short-term forecasting of intersection traffic turning movement flows (TMFs) and quantification of forecasting uncertainty. The DeepAR model, a powerful deep learning framework, was harnessed for its ability to model a large number of time series together and provide probabilistic forecasting. Through an extensive set of computational experiments, we gained valuable insights into the performance characteristics of DeepAR for TMFs forecasting, and the contributions of covariates in enhancing its forecasting performance.

The computational analysis was conducted using a large dataset covering 15 months of TMFs observations from 39 intersections in the City of Milton, Ontario. We first compared its performance with the state-of-the-art model, Transformer, and baseline autoregressive statistical model ARIMA. The

results showed that the DeepAR model outperformed Transformer and ARIMA in terms of forecasting accuracy for multiple steps forecasting, with lower RMSE and RMSPE. Furthermore, the DeepAR model provided reliable and sharp confidence intervals, which are crucial for quantifying uncertainty in traffic forecasting. To further explore the impact of covariates on forecasting accuracy, we trained two DeepAR models—one with covariates and one without. The results revealed that the model with covariates achieved better accuracy and lower uncertainty, confirming the importance of incorporating covariates in TMFs forecasting. We also investigated the effectiveness of using a shared representation layer in the DeepAR model. By processing information from multiple time series in parallel, the shared representation layer allowed the model to extract common features and cross correlation from different groups of time series data. This approach significantly improved the model's ability to capture and leverage cross-sectional features, leading to more accurate and robust predictions. In the final set of experiments, we fine-tuned the DeepAR model using a smaller dataset of five hold-out intersections. The results demonstrated that even with limited training data, the DeepAR model could still achieve competitive performance, providing valuable insights for the model's practical applicability in real-world scenarios.

Overall, this research presents significant contributions to the field of real-time short-term intersection traffic turning movement flows prediction and uncertainty quantification. DeepAR model is capable to handle multiple time series, time-varying covariates, and cold start scenarios, that makes it a powerful tool for traffic management applications. Meanwhile, the reliably proofed forecasting confidence interval by DeepAR has the potential to enhance traffic management and strengthen traffic performance. The findings shed light on the importance of considering turning movement groups and covariates for more robust and reliable TMFs forecasting. The comprehensive evaluation of prediction performance and uncertainty quantification provided valuable insights for traffic management schemes, ensuring optimal traffic control and efficient decision-making.

For future research, there are several promising directions to consider. Firstly, the DeepAR model still encounters challenges in capturing extreme cases such as rapid growth during peak hours. This limitation could potentially be addressed by integrating different mechanisms such as Wavelet analysis(Farge, 1992) or minimizing quantile loss(Koenker, 2005) in the future. Then, conducting an in-depth analysis of the training data to unveil underlying patterns could offer insights into how dataset size and distribution influence forecasting errors and uncertainties. Additionally, exploring the transferability

of the model to intersections in different cities could be crucial for reducing the similarities between training and hold-out intersections. Furthermore, the inclusion of additional covariates, such as varying intersection geometries, neighborhood land use, weather conditions, and special events, could enhance the model's accuracy and robustness.

Chapter 5 Temporal Fusion Transformer for Real-Time Intersection Turning Movement Flow Forecasting Incorporating Exogenous Factors

Chapter 4 explored intersection TMFs forecasting through the DeepAR model, focusing on generating probabilistic predictions and reliable confidence intervals. While DeepAR significantly advanced traffic forecasting by capturing cross-sectional features and external covariates, its primary emphasis was on producing probabilistic forecasts rather than explicitly highlighting the importance of uncertainty quantification or dissecting the specific impacts of exogenous factors—such as road geometry, weather, or land use—on traffic dynamics. This limitation underscores a broader challenge in traffic forecasting: understanding how and why specific variables influence traffic patterns, essential for creating interpretable and adaptable models suitable for real-world deployment.

This chapter addresses this gap by introducing the Temporal Fusion Transformer (TFT), a state-of-the-art deep learning framework designed to integrate historical TMFs data, static covariates (e.g., speed zones, road types), and time-dependent inputs (e.g., weather, holidays). Building on Chapter 4's probabilistic foundation, the TFT model not only quantifies uncertainty but also provides interpretable insights into the relative importance of exogenous factors. By leveraging attention mechanisms and variable selection networks, TFT identifies which covariates—such as hour-of-day or road speed limit—most significantly impact traffic flow, enabling targeted improvements in forecasting accuracy.

5.1 Literature review

As discussed in the previous chapters, researchers have dedicated significant efforts to developing statistical and machine learning models for traffic forecasting. These models aim to capture the temporal dependencies inherent in traffic flow data. For example, Alghamdi et al. (Alghamdi et al., 2019) employed the Autoregressive Integrated Moving Average (ARIMA) model for daily traffic flow forecasting. Their results demonstrated the effectiveness of ARIMA in capturing trends and seasonality from historical time series data. However, for real-time forecasting (minutes to hours), the nature of TMFS data becomes more complex. TMFS data often exhibits significant non-stationarity, exceeding the capabilities of ARIMA models. To address the challenges posed by non-stationary TMFS and to capture their complex non-linear time-dependent features, machine learning models have gained popularity. Recent studies have explored various approaches such as K-Nearest Neighbors (K-NN) regression (Sun et al., 2018), Support Vector Regression (SVR) (Wei & Liu, 2013), Random Forest (RF) (Liu & Wu, 2017), Long Short-Term Memory (LSTM). For Example, a comparative study by Smith and Demetsky (Smith & Demetsky, 1997), K-NN outperformed ARIMA in forecasting 15-minute traffic flow on two highways in Virginia, with K-NN showing an average absolute error 18% lower than ARIMA. Similarly, Deshpande et al. (Deshpande & Bajaj, 2016) demonstrated the effectiveness of SVR in short-term traffic forecasts, achieving a Root Mean Square Error (RMSE) of only 0.42 vehicles per minute for 1-minute aggregation interval forecasting.

In addition to the traditional ML models, many researchers have recently attempted to apply various deep learning models for traffic forecasting. Zhang et al. (Zhang, Pan, & Fu, 2024) developed a Parallel Bidirectional LSTM (PB-LSTM) model specifically designed to make forecasting based on prior-known future time-dependent inputs. These inputs included factors like hour of day, day of week, holidays, and historical TMFs. Their results demonstrated that the PB-LSTM model outperformed a standard LSTM model, particularly in terms of transferability for forecasting TMFS at new intersections. This highlights the potential benefits of incorporating such time-dependent external factors into traffic flow forecasting models. Additionally, Li et al. (Li, Ban, et al., 2020) proposed a novel Deep Intersection Spatial-Temporal Network (DISTN) that combined LSTM and CNN models for TMFs forecasting. Their study utilized VISSIM simulation data to evaluate the performance of DISTN in 8 intersections with varying combinations of external factors. These factors included aspects like weekdays versus weekends and peak versus off-peak hours. By incorporating these external factors into the model, they observed a significant improvement in forecasting accuracy. Their study specifically reported a decrease in Mean

Absolute Percentage Error (MAPE) from 18.48% to 16.74% for 15-minute intervals. This finding further emphasizes the value of considering external factors when developing TMFs forecasting models.

Another deep learning model which has gained significant popularity due to its success in Large Language Model (LLM) is the transformer models (Vaswani et al., 2017). Transformer based models have found success in diverse domains such as stock price forecasting(Hu, 2021), demand forecasting(Nazir et al., 2023), and electricity load forecasting(Huy et al., 2022). Additionally, several studies have explored the application of transformer models in traffic volume forecasting, and their results have shown that it outperformed classical LSTM models by at least 10% in accuracy on their datasets (Cai et al., 2020; Grigsby et al., 2021; Yan et al., 2021).

While these studies have successfully combined exogenous factors with historical traffic turning movement flow to enhance forecasting accuracy, a notable limitation of these studies is their concentration on a restricted range of exogenous factors, neglecting to analyze the performance of Temporal Fusion Transformer (TFT) models across various exogenous factors and a few key questions still remain to be addressed: (1) What is the magnitude of the effect of various exogenous factors in terms of improving forecasting accuracy, particularly for multi-step forecasting? (2) Which time-dependent exogenous factors (e.g., weather, time of day) have the most significant impact on forecasting accuracy? (3) What is the relative importance of the various static covariates (e.g., speed zone, road type) for different turning movements? A deeper understanding of these relationships is crucial for building targeted forecasting models that leverage the combined power of historical data and external influences.

5.2 Research design and methodology

5.2.1 Forecasting framework

This study leverages the Temporal Fusion Transformer (TFT) model (Lim et al., 2021) for real-time forecasting of future TMFs. It integrates a mechanism for evaluating the influence of exogenous factors, offering interpretability for understanding the factors shaping traffic flow. The model incorporates several novel architectural elements to achieve these goals: 1) static covariate encoders, which encode static features (e.g., speed zone, road type) for use throughout the network; 2) gating mechanisms and sample-dependent variable selection network to minimize the influence of irrelevant inputs, focusing the

model on the most impactful factors; 3) sequence-to-sequence layer which processes known and observed inputs (e.g., historical TMFS) locally, capturing short-term dependencies; and 4) temporal self-attention decoder which learns long-term dependencies within the dataset, identifying patterns that influence future traffic flow.

Figure 5.1 shows the data processing and model training pipeline for utilizing the Temporal Fusion Transformer (TFT) model in intersection traffic TMFs forecasting. The process begins with data extraction. Here, three key components are extracted from the raw data: historical TMFS data, static covariates (e.g., speed limit, road type), and prior-known future time-dependent covariates (e.g., weather, holidays). Next, static and time-dependent covariates undergo preprocessing to ensure their suitability for the TFT model. This involves normalization and scaling of the data. Additionally, a correlation analysis is conducted to identify and remove redundant variables that might hinder model performance. After preprocessing, the data is formatted into a structure compatible with the TFT model. We then employ a random search method (Liaw et al., 2018) to identify the optimal hyperparameter settings for the TFT model, maximizing its performance on this specific dataset. The preprocessed data might undergo an additional step of data aggregation. This step transforms the time intervals into a format suitable for the model's requirements. The forecasting horizon is also set at this stage. After all these steps, the TFT model undergoes training on the historical TMFs data. Upon inference, the trained TFT model generates predictions for future TMFs. The quality and accuracy of these predictions are then rigorously analyzed to assess the effectiveness of the overall forecasting framework.

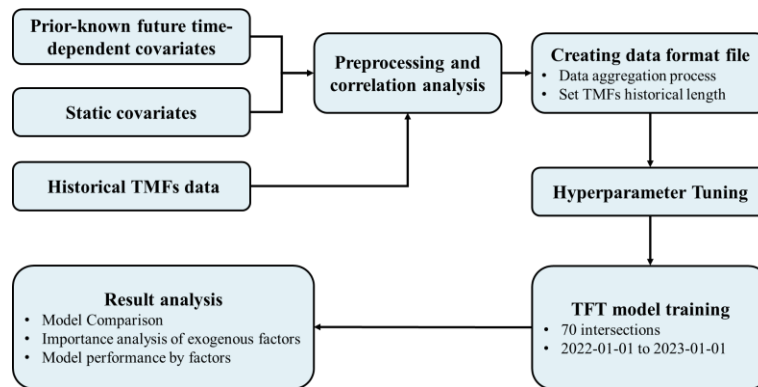


Figure 5.1: Framework of utilizing the TFT for TMFs forecasting

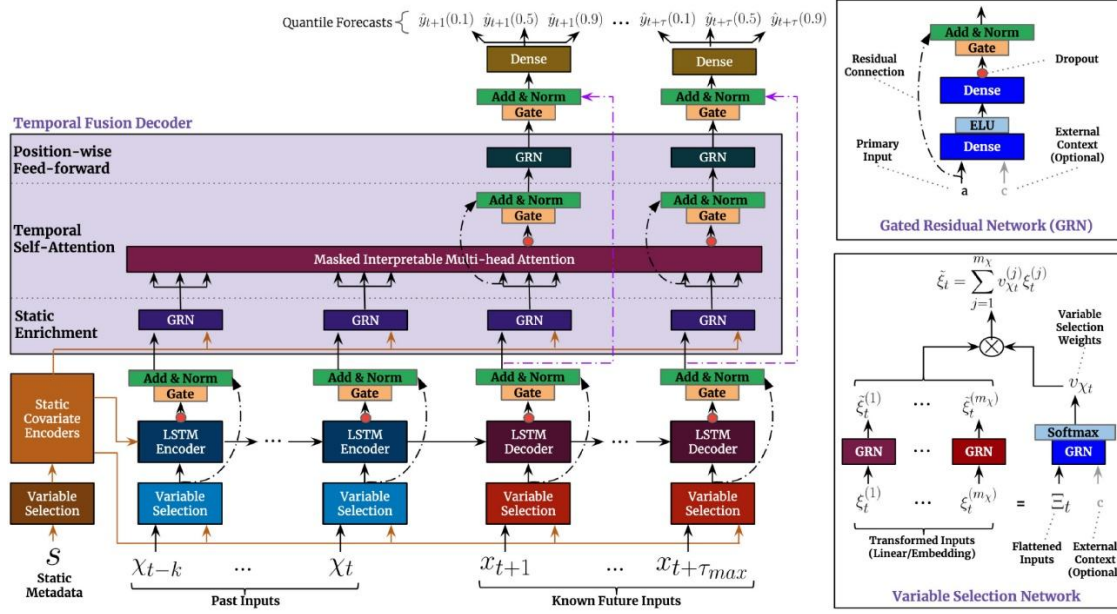


Figure 5.2: The structure of Temporal Fusion Transformer(Lim et al., 2021).

5.2.2 Temporal Fusion Transformer architecture

In this research, we examine a TMFs dataset comprising 12 movements, each associated with a set of static covariates $s_i \in R^{m_s}$, as well as prior- known time dependent input $X_{i,t} \in R^{m_x}$ and target $y_{i,t} \in R$ at each time step $t \in [0, T_i]$. The time dependent input are divided into two parts $X_{i,t} = [z_{i,t}^T, x_{i,t}^T]^T$, where observed TMFS $z_{i,t} \in R^{m_z}$ can only be measured at each step and are unknown beforehand, and known inputs $x_{i,t} \in R^{m_x}$ which can be predetermined (e.g. the day-of-week at time t). This study aimed to use static covariates s_i , observed TMFs $z_{i,t-k:t}$, and known inputs $x_{i,t-k:t+\tau}$ to make the prediction of $y_{i,t:t+\tau}$.

To forecast $y_{i,t}$, TFT models employ multiple architectures as depicted in Figure 5.2. The main components of TFT include:

1. Gated Residual Network (GRN): This serves as the basic cell unit for other TFT components, designed to ignore unnecessary components and provide adaptive depth and network complexity suitable for large datasets and different scenarios.
2. Variable Selection Network (VSN): These are used to select highly relevant input variables at each time step.

3. Static Covariates Encoders: These combine static features with temporal features to enhance model robustness and transferability.
4. Temporal Processing Model: This includes a Sequence-to-Sequence layer for capturing short-term temporal dependencies and a Multi-head Attention module for long-term correlations.

5.2.2.1 Gated Residual Network (GRN)

In the TFT model, a GRN was proposed as the basic block to give the model flexibility in the process of capturing the input features. The input of GRN includes two vectors: the first one is the primary information vector represented as a and the second one is the context information vector represented as c . The calculation process of GRN is defined in Equation (5.1),

$$GRN_w(a, c) = LayerNorm(a + GLU_w(\eta_1)) \quad (5.1)$$

Where η_1 and η_2 are the outputs of two intermediate layers as in Equation (5.2) and Equation (5.3).

$$\eta_1 = W_1\eta_2 + b_{1,w} \quad (5.2)$$

$$\eta_2 = ELU(W_{2,w}a + W_{3,w}c + b_{2,w}) \quad (5.3)$$

Where η_1 and η_2 are the outputs of two intermediate layers as in Equation (5.2) and Equation (5.3); LayerNorm is a standard layer normalization function (Ba et al., 2016) and ELU is Exponential Linear Unit activation function (Clevert et al., 2015).

The GLU (Dauphin et al., 2017) is Gated Linear Units as shown in Equation (5.4), which is used as a key to control how GRNs can contribute to the input.

$$GLU_w(\gamma) = \sigma(W_{4,w}\gamma + b_{4,w}) \odot (W_{5,w}\gamma + b_{5,w}), \quad (5.4)$$

$\sigma(\cdot)$ represents the sigmoid activation function, $\odot$ represent the elementwise Hadamard product, $W_{(\dots)}$ and $b_{(\dots)}$ represent weight and biases.

5.2.2.2 Variable Selection Network (VSN)

VSN is used to select the most relevant input variables for the TFT model. This eliminates the influence of variables that are useless to improve forecasting accuracy. Entity embeddings were used to represent categorical variables. The calculation process of variable screening weights is shown in Equation (5.5) and Equation (5.6).

$$v_{x_t} = \text{softmax}(\text{GRN}_{v_x}(\Xi_t, c_s)), \quad (5.5)$$

$$\tilde{\xi}_t = \sum_{j=1}^{m_x} v_{x_t}^{(j)} \xi_t^{(j)}, \quad (5.6)$$

where v_{x_t} is variable selection weights, c_s is output of static covariate encoder, and Ξ_t is the flattened vector of all transformed past inputs at time t , equal to $[\xi_t^{(1)T}, \dots, \xi_t^{(m_x)T}]$. $\xi_t^{(j)}$ is the output of gated residual network of variable j and $\tilde{\xi}_t$ is the final output of VSN. The VSN is a crucial component of the TFT model, providing interpretable insights into the significance of covariates by calculating their variable selection weights.

5.2.2.3 Static covariate encoders

Compared with previous time series forecasting models, the TFT model also uses information from static input variables. It employed different GRN encoders to generate four kinds of context vectors, c_s , c_e , c_c , and c_h .

These different kinds of context vectors were used in different places in the temporal fusion decoder. Static information plays an important role in these places: (1) temporal variable selection (c_s); (2) locally process temporal features (c_c , c_h); (3) enrichment of temporal features (c_e).

5.2.2.4 Interpretable multi-head attention

In TFT models, an interpretable multiple head attention (S. Li et al., 2019) structure is applied for each head to share weights (Equation (5.7) and Equation (5.8)).

$$\text{InterpretableMultiHead}(Q, K, V) = \tilde{H}W_H, \quad (5.7)$$

$$\tilde{H} = \left\{ \frac{1}{m_H} \sum_{h=1}^{m_H} A(QW_Q^{(h)}KW_K^{(h)}) \right\} VW_V, \quad (5.8)$$

where $A(\cdot)$ is Normalization function, V is scale values, K is keys and Q is queries (Vaswani et al., 2017) as same as attention mechanism. $W_K^{(h)}$, $W_Q^{(h)}$ and W_V are head-specific weights for keys, queries, and values, and W_H linearly combines outputs concatenated from all heads H_h .

Compared with standard attention mechanism (Vaswani et al., 2017) and general multi-head attention, the interpretable multiple head attention mechanism can capture the longer temporal dependency and provide the indicative of a particular features' importance.

5.2.2.5 Temporal fusion decoder

Building upon the multi-head attention module's ability to capture long-term dependencies, TFT incorporates a temporal fusion decoder. This decoder utilizes a series of specialized layers (described below) to further enhance its ability to learn and exploit the temporal relationships present within the dataset.

a. Sequence to sequence layer

In traffic time series data, traffic flow of one movement at time t is often related to the flows of other surrounding movements (Chen et al., 2017). To leverage this local context, TFT model applied a sequence-to-sequence layer which feed $\tilde{\xi}_{t-k:t}$ into the encoder and $\tilde{\xi}_{t+1:t+\tau}$ to decoder for generating a set of uniform temporal feature as next inputs $\phi(t, n)$ with n being a position index into the temporal fusion decoder itself. Meanwhile, to allow static covariates to influence local processing, the TFT uses the temporal features c_c , c_h from the static covariate encoders to initialize the state of the first encoder unit. A gating layer is applied here to avoid redundant connections:

$$\tilde{\phi}(t, n) = \text{LayerNorm}(\tilde{\xi}_{t+n} + \text{GLU}_{\tilde{\phi}}(\phi(t, n))), \quad (5.9)$$

b. Static enrichment layer

To enhances the temporal features with static features, TFT applies a static enrichment layer, which takes the form defined by Equation (5.10) for a given position index n :

$$\theta(t, n) = GRN_{\theta}(\tilde{\phi}(t, n), c_e), \quad (5.10)$$

where c_e is the context vector from a static covariate encoder.

c. Temporal self-attention layer

Following the static enrichment layer is a self-attention layer, in which all the static enriched temporal features are first grouped into a single matrix $\Theta(t) = [\theta(t, -k), \dots, \theta(t, \tau)]^T$, and then an interpretable multi-head attention is applied at each forecast time:

$$B(t) = InterpretableMultiHead(\Theta(t), \Theta(t), \Theta(t)) \quad (5.11)$$

After that, a decoder mask is applied to ensure that each temporal dimension can only attend to features. Furthermore, an additional gating layer is also applied to facilitate training:

$$\delta(t, n) = LayerNorm(\theta(t, |n) + GLU_{\delta}(\phi(t, n))) \quad (5.12)$$

d. Position wise feed forward layer

The output of the self-attention layer is then processed by an additional non-linear processing layer by GRNs:

$$\Psi(t, n) = GRN_{\psi}(\delta(t, n)) \quad (5.13)$$

As in the case of other temporal fusion decoder layers, a gated residual connection which skips over the entire transformer block, providing a direct path to the sequence-to-sequence layer— yielding a simpler model as shown below:

$$\tilde{\Psi}(t, n) = LayerNorm(\tilde{\phi}(t, |n) GLU_{\delta\tilde{\Psi}}(\Psi(t, n))) \quad (5.14)$$

e. Quantile outputs

Instead of only providing point predictions, the TFT model generates prediction with confidence intervals. This is achieved by the simultaneous prediction of various percentiles (e.g. 10th, 50th and 90th) at each time step. Quantile forecasts are generated using a linear transformation of the output from the temporal fusion decoder:

$$\hat{y}(q, t, \tau) = W_q \tilde{\Psi}(t, n) + b_q, \quad (5.15)$$

where W_q, b_q are linear coefficients for quantile q .

5.3 Experimental results

5.3.1 Study area and data description

A case study is conducted to demonstrate the application of the proposed TFT model for intersection TMFs forecasting. The case study utilizes data from the Region of Waterloo, Ontario, Canada, where a total of 211 signalized intersections have instrumented with Miovision's SmartSense systems, a video-based detection system offering continuous turning movement counts. For this study, 76 intersections, which have over 507 days of data, were selected for forecasting, as shown in Figure 5.3 (a) and the distribution of the selected intersections. Our dataset is comprised of three key components:

- **Past Observed TMFs (endogenous factors):** These represent historical turning movement counts at each intersection at an interval of 15 minutes.
- **Prior-Known Future Time-Dependent Inputs (exogenous factors):** These variables are known in advance and can influence traffic flow. Examples include the hour of the day and temperature. Data for these inputs was sourced from Environment Canada.
- **Static Covariates (exogenous factors):** These features remain constant over time and include information such as speed zone and road type. Data for these covariates was obtained from the Region of Waterloo Open Data program (Waterloo, 2022).

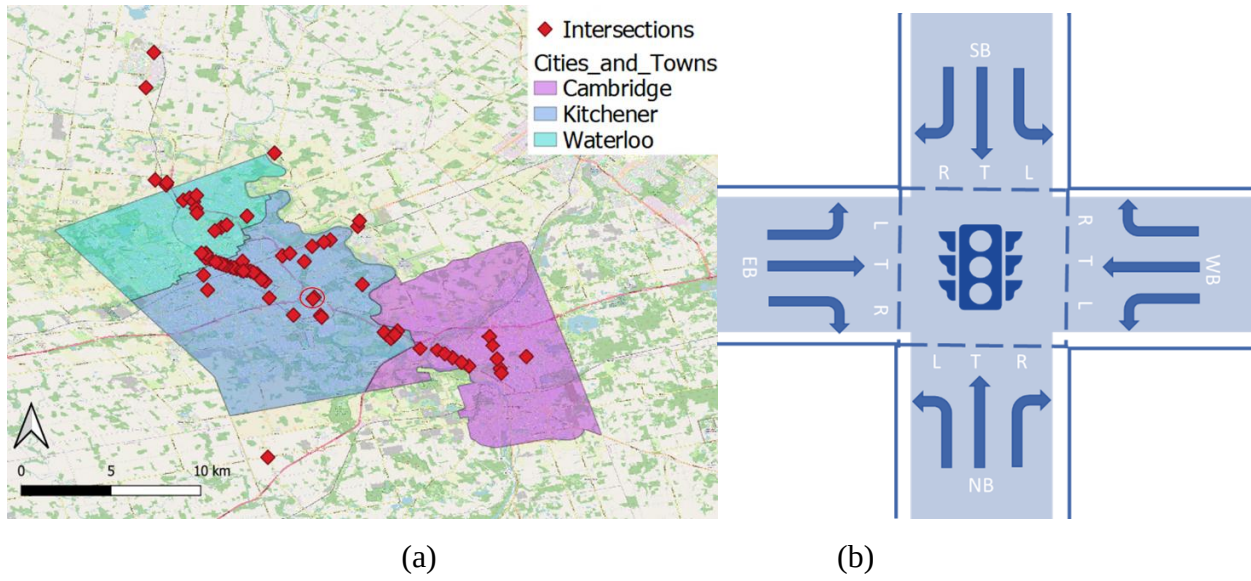


Figure 5.3: Intersection's location(a) and turning movements(b).

5.3.1.1 TMFs dataset

The TMFs dataset includes traffic counts for all turning movements at a 1-minute interval, as depicted in Fig. 3 (b). For the 76 intersections, the dataset spans from January 1st, 2022, to May 22nd, 2023. The dataset was first aggregated into 15-minute intervals to mitigate noise, which yields 48,672 records for each intersection, resulting in a total of 3,699,072 records for the entire dataset. For training purposes, data from 70 intersections spanning January 1st, 2022, to March 1st, 2023, were utilized. Testing, on the other hand, employed data from the same intersections but from March 1st, 2023, to May 22nd, 2023, and other six hold-out intersections with data from January 1st, 2022, to March 1st, 2023.

5.3.1.2 Prior-known future time-dependent covariates

In addition to the TMC data itself, time-dependent covariates provide valuable insights into future conditions that can affect traffic flow and play a significant role in influencing traffic turning movement flow. In this research, we not only incorporate timestamp labels such as the day of the week, hour of the day, and month of the year to capture recurring patterns, but also consider other complex time-varying covariates. We extracted weather related time-varying covariates and conducted a Spearman's rank correlation test (Zar, 2005) to eliminate redundant or overlapping covariates. Table 5.1 provides a detailed overview of all the selected time-dependent covariates, these include holidays, visibility, weather descriptions, precipitation amounts and temperature. By including these factors, we aim to enhance the predictive accuracy of our models by accounting for various environmental and temporal influences on traffic flow dynamics.

Table 5.1: Prior-Known Future Time-Dependent Inputs

Variable Type	Variables	Description
Time Stamp	Hour	Hour of the day (24-hour format)
	Holiday	Public holiday in Ontario
	Weekday	Day of the week (e.g., Monday, Tuesday, etc.)
	Month	Month (e.g., January, February, etc.)
	Year	Year
Weather	Visibility	Road visibility(100m)
	WeatherDescr	Weather description in English
	PrecipAmount	Amount of precipitation(cm)
	TEMP	Temperature (°C)

5.3.1.3 Static covariates

Static covariates, such as road and intersection layout, as well as intersection locations, are expected to be crucial for forecasting TMFs. Intersection layout factors like speed zones, road types, flow directions, and the number of lanes may affect traffic flow and capacity. Meanwhile, intersection locations, including surrounding land use types and municipal areas, could also affect traffic demand. These static variables collectively shape the traffic dynamics at intersections and should therefore be tested for their effect on TMFs forecasting. Table 5.2 provides a detailed overview of all the selected static covariates. Same as time-dependent covariates, we have conducted correlation analysis of raw static covariates, and applied Density-based spatial clustering of applications with noise (DBSCAN) (Khan et al., 2014) to clustering intersections based on their locations (longitude and latitude).

Table 5.2: Static Covariates

Variable Type	Variables	Description
Road and Intersection	RoadClass	Road category (Arterial, Collector, Local Street, Ramp)
	FlowDirection	Direction (Northbound, Southbound, Eastbound, Westbound)
	SpeedZone	Posted Speed limit
	DividedRoad	Indicates if the road is divided or not
	NumberofLanes	Number of lanes on the road
Location	LandUse	Land use category (Assembly, Business, Educational, Industrial, Institutional, Mercantile, Residential, missed)
	Municipality	Belonged city (Kitchener, Waterloo, Cambridge)
	DBSCAN	Location clustering

5.3.2 Experimental set-up

The TFT models were implemented using a PyTorch backend (Paszke et al., 2019). To ensure model robustness, cross-validation was applied at each iteration, with 20% of the training dataset randomly selected for model validation. The model's hyperparameter was determined by the built-in Ray Tune (Liaw et al., 2018) from Pytorch Lightning (Falcon, 2019). Based on the preliminary results that compared different historical data ranges as input, it was found that utilizing a 24-hour (96 intervals) historical input data range proved to be achieved the best results in terms of accuracy. The average training time for forecasting the next 15 minutes (1 interval) was approximately 11 seconds per iteration per training batch (batch size = 2048 records) on a NVIDIA GeForce V100. During testing on the test dataset, the forecasting process was completed within 4.30 hours, implying that each prediction was performed in under 1.25×10^{-3} seconds which makes real-time forecasting possible. The subsequent section presents the key findings from various computational experiments.

5.3.3 Result analysis and comparison

Figure 5.4 showcases an example of TMFs forecasting on the south bound and east bound approaches generated by the TFT model for the intersection of Fairway at Wilson (circled in Fig. 5 (a)) on Tuesday, May 2nd, 2023. The forecasting period spans from 4:00 AM to 11:00 PM. The model utilizes historical TMFS data for the preceding 24 hours (represented by 96-time intervals) and generates predictions at 15-minute intervals (1 time interval per prediction). The results demonstrate the TFT model's ability to accurately capture the overall trend of TMFs for all turning movements at this intersection. Notably, the model successfully identifies the two distinct peak periods observed in the "go through" (T) movements, effectively capturing the rapid increase in traffic flow during these times. However, the model tended to underestimate the traffic volume during peak periods, which is a limitation to be further investigated.

Table 5.3 presents a comparative analysis of the multistep forecasting performance among different models, including those with and without exogenous covariates, for TMFs forecasting. The evaluation is based on two key metrics: Root Mean Squared Error (RMSE) and Root Mean Squared Percentage Error (RMSPE). Two variation of the TFT models, namely, TFT with covariates (TFT_co) and TFT without covariates (TFT_non_co), are trained and compared to the models discussed in the previous chapters, including: DeepAR with covariates (DeepAR_co), DeepAR without covariates (DeepAR_non_co), and LSTM.

Table 5.3 also spotlights TFT with exogenous covariates (TFT_co) as being superior in forecasting accuracy, consistently achieving the lowest errors (RMSE & RMSPE) across all prediction steps. While all models predictably exhibit increasing errors (RMSE & RMSPE) as the forecasting horizon expands (1 to 96 steps), TFT_co maintains its dominance. DeepAR models (with/without covariates) trail closely, demonstrating competitive performance.

Interestingly, while TFT_non_co model have slightly higher RMSE values compared to the DeepAR models, their lower RMSPE values indicate a strength in capturing peak TMFS values or extreme fluctuations within the data. LSTM models, though performed reasonably well, lag behind, particularly in multi-step forecasting scenarios.

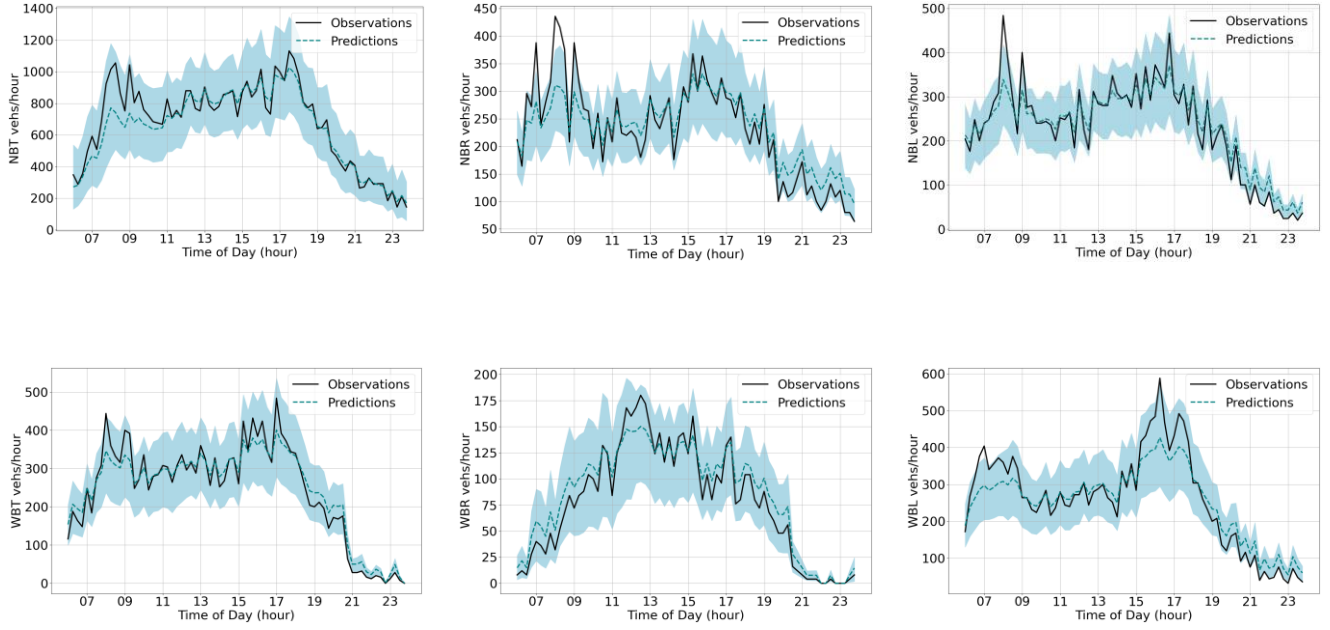


Figure 5.4: Forecasting results of TFT on May 2nd, 2023, at intersection Fairway at Wilson.

Table 5.3: Forecasting results of different models

	TFT_co		TFT_non_co		DeepAR_co		DeepAR_non_co		LSTM	
Steps	RMSE	RMSPE	RMSE	RMSPE	RMSE	RMSPE	RMSE	RMSPE	RMSE	RMSPE
	(vehs/h)	(%)	(vehs/h)	(%)	(vehs/h)	(%)	(vehs/h)	(%)	(vehs/h)	(%)
1(15 mins)	9.16	5.46	13.27	7.9	11.41	8.98	11.78	10.56	17.89	9.77
4(1 hour)	12.55	7.91	18.92	13.62	14.23	12.9	15.34	14.14	20.86	9.76
12(3 hours)	13.07	10.74	19.78	17.51	18.1	23.56	20.17	24.22	23.67	11.94
48(12 hours)	15.98	14.03	24.64	24.18	19.74	29.56	22.9	26.21	24.78	29.79
96(24 hours)	19.14	23	29.9	31.8	19.71	30.19	22.12	35.36	25.78	37.04

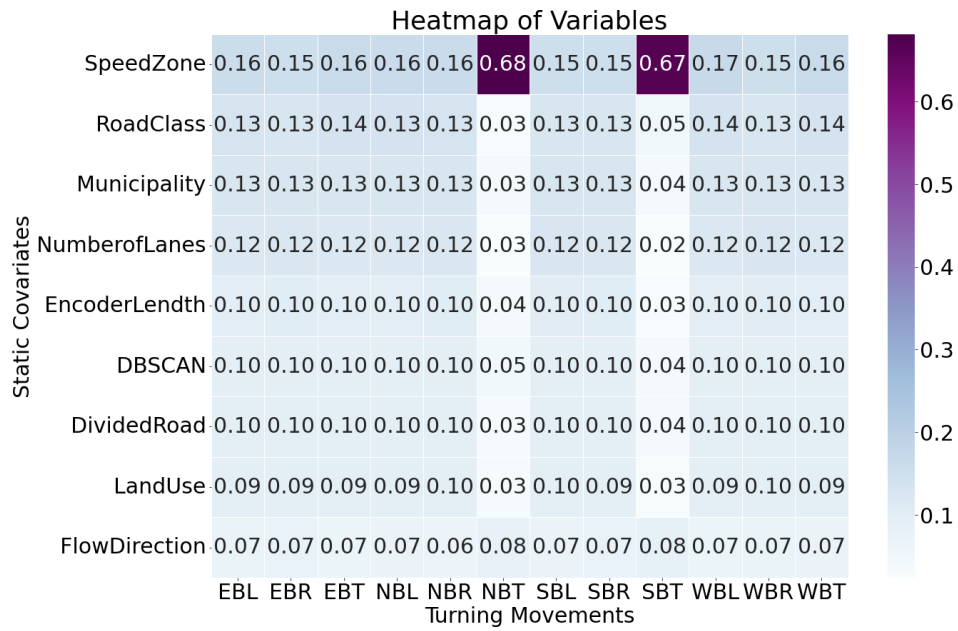


Figure 5.5: Importance of static covariates for each movement

Figure 5.5 and Figure 5.6 demonstrate the importance of static covariates and time-dependent covariates for 1 step (15-minute) on test dataset. Figure 5.5 underscores the importance of road layout factors in traffic forecasting. Speed limits, road types, and the number of lanes all significantly influence traffic flow patterns. Higher speeds tend to correlate with increased traffic volumes, particularly for "go through" movements traveling north-south. This makes sense given that our study area's intersections are primarily located along north-south corridors. Additionally, road characteristics like category and lane count directly impact traffic capacity, especially during congestion. Beyond capacity, factors like flow direction and land-use indicators (Land use, municipal categories) influence traffic demand and flow dynamics. By incorporating these road layout features, traffic forecasting models can learn the varying capacities and flow patterns of different road segments, ultimately leading to more accurate and informative forecasting.

Figure 5.6 (a) displays the most critical factors influencing traffic forecasting through a variable importance analysis. The analysis reveals that the count of historical TMFs observations emerges as the single most influential factor. This aligns with findings from Lim et al. (Lim et al., 2021) in time series forecasting, where historical data plays a crucial role. Following closely is the relative time index, highlighting the importance of the temporal order within the historical observations. Beyond these core

factors, the prominence of the hour feature emphasizes the significance of temporal patterns in traffic flow. The inclusion of year, month, and weekday variables further underscores the importance of considering seasonal and weekly variations that influence traffic behavior. Weather also plays a significant role, with temperature, visibility, weather description, and precipitation amount all holding notable importance. This highlights the impact of weather conditions on traffic flow. Additionally, the inclusion of the holiday status feature provides valuable context for understanding deviations from typical traffic patterns.

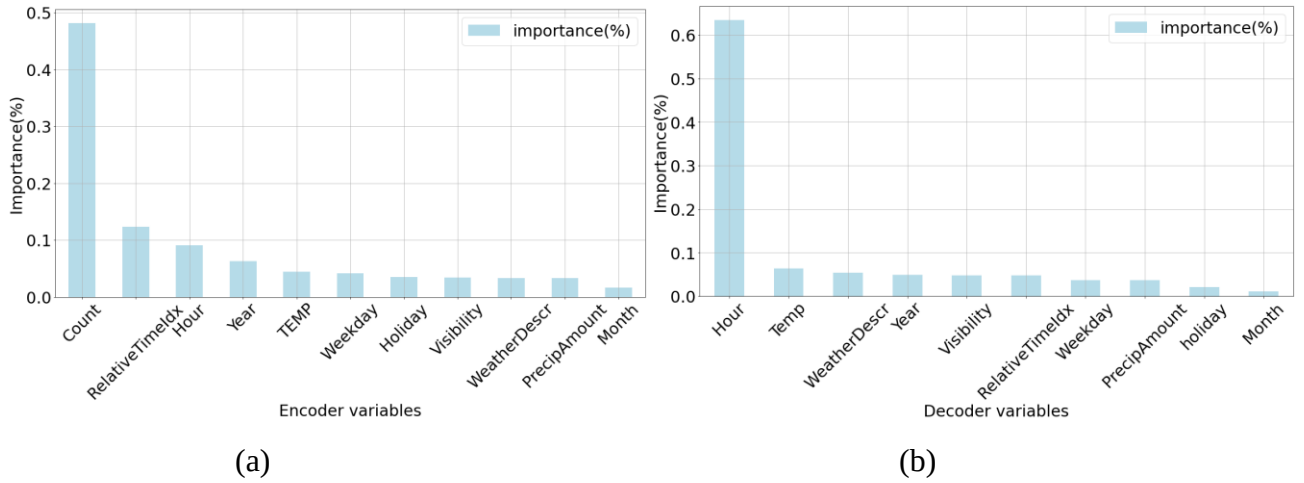
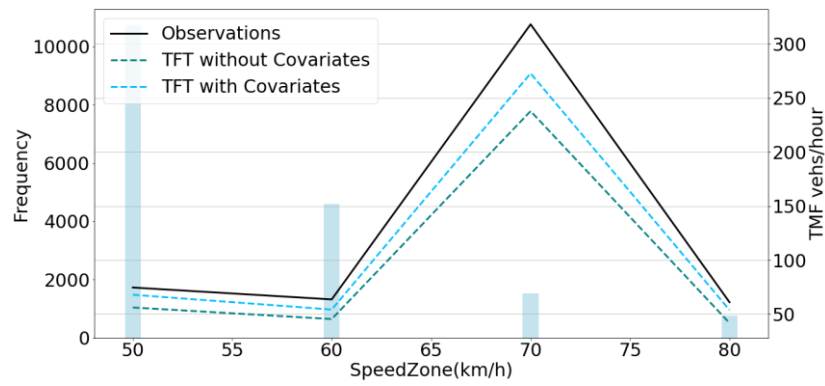


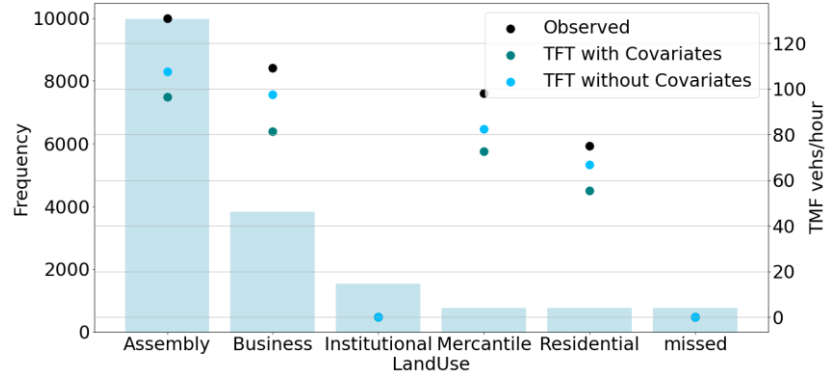
Figure 5.6: Importance of time-dependent covariates for learning (a) and forecasting (b)

Figure 5.6 (b) shows the time-based factors shaping traffic flow forecasting within the TFT model. Here, "hour" reigns supreme, underlining its significant influence on traffic patterns. Weather also plays a critical role, with temperature, weather description, visibility, and precipitation all featuring prominently. This underscores the importance of considering weather conditions for accurate forecasting. Interestingly, seasonal (year, month) and weekly (weekday) variations hold less importance compared to weather, suggesting that weather might be a more dynamic and impactful factor in this specific study area.

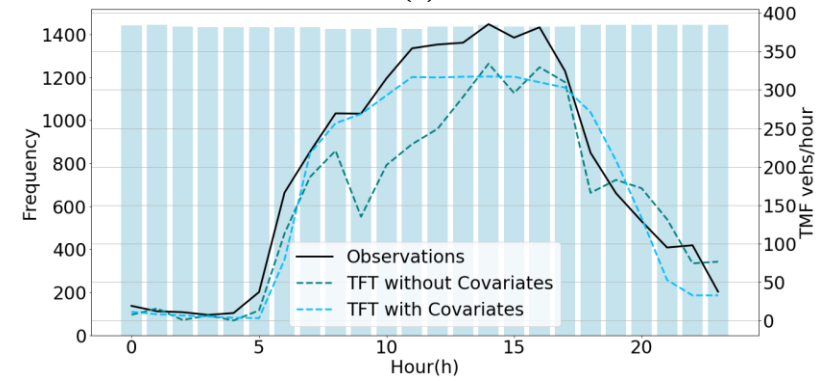
To evaluate the TFT model's performance under the influence of various exogenous factors, we segmented the test dataset based on several exogenous factors: speed zone, land use, hour, and precipitation amount (100-meter increments). We then compared the performance of the TFT model with covariates (TFT_co) to the model without covariates (TFT_non_co).



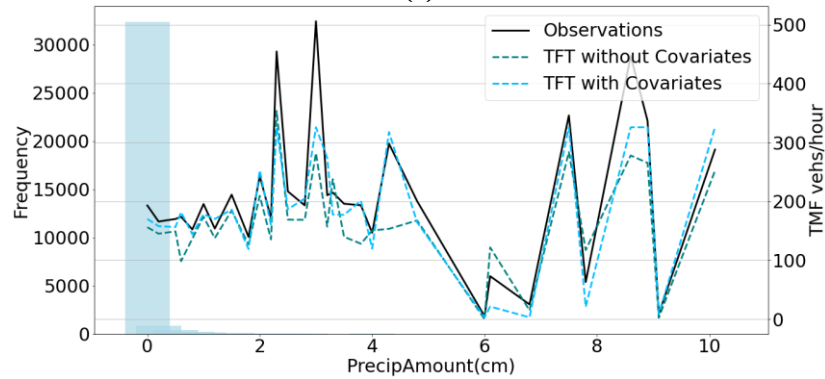
(a)



(b)



(c)



(d)

Figure 5.7: TFT forecasting performance based on exogenous factors.

Figure 5.7 presents histograms depicting the distribution of each factor's categories, along with scatter plots comparing predicted TMFs (by both models) to the actual observed TMFs. The analysis suggests that higher traffic flow can negatively impact the performance of the TFT_non_co model. This highlights the importance of incorporating covariates, as TFT_co consistently outperforms TFT_non_co across all scenarios. Even under varying external factors (time-varying covariates), TFT models benefit from including covariates. However, when traffic flow is high, both models exhibit greater difficulty in making accurate forecasting due to increased uncertainty.

5.4 Conclusions

This research has investigated the application of Temporal Fusion Transformer (TFT)(Lim et al., 2021) models for intersection movement-level traffic flow forecasting, focusing on how exogenous factors improve prediction accuracy. The research presents three principal contributions:

- Firstly, it introduces a forecasting pipeline utilizing TFT models for TMFs forecasting, integrating historical TMFs data, static covariates, and future time-dependent inputs. Furthermore, we conduct a comparative analysis of this framework's performance against other prevalent deep learning models used in TMFs forecasting.
- Secondly, to delve deeper into the nuances of traffic flow determinants, we undertake a thorough factor importance analysis. This examination unveils the pivotal exogenous factors exerting the most substantial influence on TFT model forecasting.
- Thirdly, by segmenting the dataset into various subsets based on specific exogenous factors, we investigate how the TFT model performs under diverse conditions, shedding light on its vulnerabilities concerning certain factors.

Chapter 6: Real-Time Traffic Turning Flow Forecasting under Few-Shot and Zero-Shot Scenarios by Decoder-Only Model

Chapter 5 demonstrated the efficacy of the Temporal Fusion Transformer (TFT) in intersection TMFs forecasting, emphasizing the significant role of incorporating exogenous factors—such as static road characteristics and dynamic weather conditions—to enhance forecasting accuracy and interpretability. However, the effectiveness of TFT diminishes in zero-shot and few-shot scenarios, where historical data for new or infrequently monitored intersections is scarce or entirely unavailable. As an attempt to address this critical limitation, this Chapter explores the application of decoder-only models, specifically TimesFM, for zero-shot and few-shot TMFs forecasting. In this effort, we attempt to further enhance TimesFM by integrating Empirical Mode Decomposition (EMD) to handle complex traffic patterns and Random Forest (RF) to incorporate external covariates, which is expected to improve forecasting robustness and adaptability in scenarios with limited or no historical data.

The remainder of this chapter is structured as follows. The background section provides an overview of existing forecasting models, highlighting their limitations in zero-shot and few-shot learning scenarios. The methodology section describes the models and experimental framework, including external covariates and decomposition techniques. The experimental results section presents case study datasets, experiment set up, and analyses across different scenarios, focusing on model comparisons, external factors, and special weather impacts. The results section discusses findings, while the conclusion section summarizes insights and suggests future research directions.

6.1 Background and review of literature

Deep learning models for TMFs forecasting can be broadly categorized into two types (Fu et al., 2023): encoder-decoder models, which specialize in zero-shot and few-shot generalization by leveraging extensive pretraining, and decoder-only models, which rely on historical data representations for structured learning but struggle in zero-shot environments (Figure 6.1).

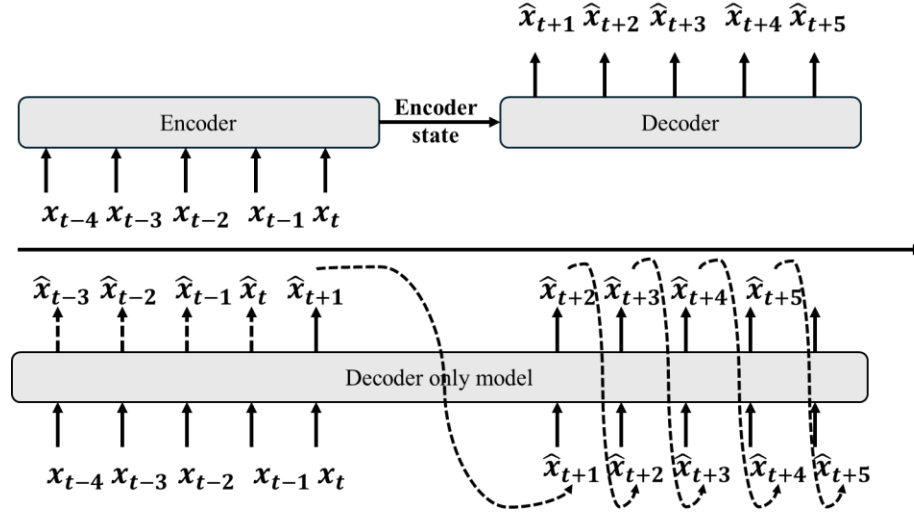


Figure 6.8: Encoder-Decoder and decoder-only time series forecasting models

6.1.1 Encoder-decoder models

Encoder-decoder models were first introduced for natural language processing (NLP) tasks (Sutskever, 2014) and were later adapted for time series forecasting. This architecture consists of two distinct stages: the encoder, which processes historical data to extract meaningful representations and capture long-term dependencies, and the decoder, which leverages these representations to generate sequential forecasts. This flexible design can be implemented using recurrent neural networks (RNNs) (Sherstinsky, 2020) such as Long Short-Term Memory (LSTM) (Kao et al., 2020) and Gated Recurrent Units (GRU) (Fu et al., 2016) or enhanced with attention-based mechanisms (Du et al., 2020) like the Temporal Fusion Transformer (TFT) (Lim et al., 2021), which improves alignment between input and output sequences, increasing robustness and performance.

In traffic forecasting, where sufficient historical data and covariates are available, encoder-decoder models have emerged as powerful tools(Fu et al., 2016; Wang et al., 2020) due to their ability to model complex, nonlinear relationships in traffic flow data. They are also highly flexible, allowing for the integration of external covariates (e.g., weather or traffic signals) and leveraging shared patterns to handle multiple related time series, such as traffic flows at different intersections. However, their reliance on rich datasets and limited generalization capabilities makes them less suitable for zero-shot scenarios(Ekambaram et al., 2024), where models must generalize to unseen locations without fine-tuning. This limitation is particularly evident in scenarios such as deploying sensors at new intersections or integrating new intersection systems, where no prior data is available for model training. In few-shot settings, encoder-decoder models can improve their performance by incorporating external covariates, such as weather conditions, intersection layouts, and traffic signal timings, which provide essential contextual information to better capture patterns in limited data scenarios. However, these models face significant challenges in scalability when applied to large-scale traffic networks or dynamic environments with rapidly changing conditions. Their adaptability is also constrained by the need for task-specific fine-tuning to generalize effectively, limiting their practicality in zero-shot scenarios. In such cases, where no prior data is available for unseen intersections or entirely new systems, encoder-decoder models often fail to provide robust and reliable predictions, highlighting the need for more flexible and scalable alternatives.

6.1.2 Decoder only models

Recent advancements in large language models (LLMs) have led to the development of decoder-only models for Natural Language Processing (NLP) and Generative artificial intelligence (GenAI) tasks(Fu et al., 2023), such as Generative Pre-trained Transformers (GPTs)(Yenduri et al., 2024). These models, which are trained on vast text datasets with encoder only architecture and autoregressive forecasting design, excel in various downstream tasks such as text generation and question answering, often in zero-shot scenarios where no task-specific fine-tuning is required. This success has inspired researchers to explore similar principles to solve challenges that occur in time series forecasting, as both NLP and time series involve sequential data with temporal dependencies and zero-shot scenarios.

In time series forecasting, decoder-only models eliminate the encoder stage entirely and adopt a single-stage architecture where the decoder directly processes historical data inputs to produce autoregressive forecasts without requiring task-specific fine-tuning. Google’s TimesFM(Das et al., 2023) is a prominent example of a decoder-only model specifically designed for time series forecasting. Drawing inspiration from NLP models like GPT, TimesFM utilizes training methodologies similar to those used for LLMs. It is pretrained on a diverse and extensive corpus of time-series data with a total of over 6 billion data points, including both synthetic data and real-world data like Google Trends, Wikipedia Pageviews, and benchmark datasets such as M4 (traffic and electricity)(Makridakis et al., 2020) and ETT(Zhou et al., 2021). This training strategy, blending 80% real-world data with 20% synthetic data, allows TimesFM to generalize across multiple domains, granularities, and temporal patterns, making it highly robust in zero-shot forecasting scenarios.

In performance benchmarks, TimesFM demonstrates state-of-the-art results in zero-shot tasks. On the Monash archive dataset(Godahewa et al., 2021), it outperformed supervised models like DeepAR(Salinas et al., 2020) and N-BEATS(Oreshkin et al., 2019) by up to 25% in scaled Mean Absolute Error (MAE).

6.1.3 Application of encoder-decoder and decoder-only models in traffic flow forecasting

The application of encoder-decoder and decoder-only models to traffic flow forecasting has been a growing area of research, particularly in addressing zero-shot and few-shot challenges. These models aim to overcome the limitations of traditional approaches in real-world scenarios characterized by data sparsity or dynamic environments. To tackle few-shot scenarios, such as new target intersections with missing data, several researchers (Fu et al., 2016; Tian et al., 2018) have applied transfer-learning methods. For instance, Zhao et al. (Z. Zhang et al., 2023) proposed an LSTM model with dynamic time warping to leverage data from similar locations to compensate for missing data at the target location. However, their model's performance was suboptimal, with an RMSE exceeding 25 vehicles per 5 minutes when 70% of the historical data was removed. Furthermore, their research did not explore the model's applicability in zero-shot scenarios, where no data is available for unseen intersections. Other studies have advanced the

field by incorporating known covariates, such as intersection location and layout, during model training (Lim et al., 2021; Salinas et al., 2020). For example, Junyi et al. (Li et al., 2021) proposed a stacked LSTM that integrated geographical attributes to provide additional contextual information, significantly enhancing prediction accuracy. Their results demonstrated that these external factors played a crucial role in improving short-term forecasting, reducing the Mean Absolute Percentage Error (MAPE) by up to 15% in scenarios with limited target data. Moreover, intersection network graph representation(W. L. Hamilton, 2020) has emerged as an important feature about the connection of the road or intersections for modeling the spatial relationships and traffic flow characteristics within a traffic network to improve model performance in both few-shot and zero-shot forecasting. For instance, Qiong et al.(Wu et al., 2021) introduced a novel cross-city deep transfer learning framework for traffic prediction with minimal data in the target city. Their model utilized a graph-based road network representation to capture spatial-temporal correlations and was trained using data from multiple known source cities. The proposed framework achieved a MAPE of 11.03% for daily traffic flow predictions in unseen target cities. Despite these advancements, graph-based models face limitations due to their reliance on predefined graph structures and sensitivity to traffic distribution shifts. These factors reduce their effectiveness in zero-shot scenarios(Ermagun & Levinson, 2018), particularly in highly diverse or complex target areas.

Compared to encoder-decoder models, decoder-only architectures exhibit significant advantages in few-shot and zero-shot scenarios due to their streamlined autoregressive design and pre-training on massive datasets. While decoder-only models have been extensively explored in NLP and generative AI tasks, their application to time series forecasting remains underexplored. Recent studies have proposed frameworks for leveraging LLMs in time series forecasting (Gruver et al., 2024; Liu et al., 2024; Rasul et al., 2023), utilizing pre-training on large datasets to enhance generalization. For example, Yong et al. (Liu et al., 2024) introduced AutoTimes, a forecaster based on LLMs, and demonstrated its strong performance in zero-shot traffic flow forecasting scenarios. Their results showed that AutoTimes, without fine-tuning, outperformed supervised encoder-decoder models like TFT and DeepAR, achieving an MAPE of 11.3%. Meanwhile, Abhimanyu et al. (Das et al., 2023) proposed the TimesFM (Das et al., 2023), which is the first decoder-only model specifically designed for time series forecasting, and their result

showed TimesFM outperformed state-of-the-art decoder-only models including AutoTimes in zero-shot forecasting scenarios while using fewer parameters.

Despite their advantages, decoder-only models like TimesFM also have limitations that hinder their effectiveness in traffic forecasting. For example, unlike encoder-decoder models, which can naturally integrate weather, road layouts, and other contextual variables, TimesFM is unable to incorporate external covariates and operates solely on historical traffic data, limiting its adaptability to real-world conditions. Moreover, TimesFM is more sensitive to noise and complex temporal patterns as it does not have an encoder to learn from the training data. While TimesFM is effective at learning general trends, traffic flow data is often influenced by short-term fluctuations, external disruptions, and varying seasonal patterns, which the model may struggle to capture effectively. To address this limitation, this research introduces a hybrid modeling approach that enhances TimesFM by integrating:

- Empirical Mode Decomposition (EMD)(Rilling et al., 2003) for addressing complex temporal patterns: As traffic flow data often contains noise and overlapping patterns that hinder accurate forecasting, EMD decomposes the traffic flow data into intrinsic mode functions (IMFs) and residuals. This approach effectively separates long-term trends, seasonal variations, and high-frequency fluctuations before feeding them into the model for prediction. This allows the model to focus on meaningful signals while minimizing noise interference, ultimately improving predictive accuracy and robustness.
- Random Forest (RF) (Rigatti, 2017) for external covariates: Traffic flow is heavily influenced by exogenous factors such as weather conditions, intersection layouts, and temporal attributes (e.g., time of day, day of the week). RF was selected due to its ability to model complex, nonlinear relationships and its robustness against overfitting, making it well-suited for integrating diverse covariates in traffic flow forecasting. Since TimesFM does not natively support covariate integration, RF is applied post-prediction to incorporate these features, refining model outputs and enhancing interpretability.

By enhancing TimesFM with EMD and RF, our study provides a scalable, interpretable, and generalizable forecasting framework for zero-shot and few-shot traffic flow prediction. The proposed approach aimed to overcome the limitations of existing models by improving

adaptability, robustness, and forecasting accuracy, particularly in real-world few-shot and zero-shot scenarios.

6.2 Methodology

In this research, the few-shot and zero-shot traffic Turning Movement Flows (TMFs) forecasting problem can be defined as build a multivariate model that takes h past intervals of TMFs time series as context and predicts p future intervals. For each movement in each target intersection, the context is denoted by $X_{m,(t-h:t)} = \{X_{m,t-h}, \dots, X_{m,t}\}$ while the observed values in the forecasting horizon are denoted $X_{m,(t+1:t+p)}$, where $m \in M$ is the turning movement. For a four-leg intersection, as depicted in Figure 6.2, there are a total of 12 possible movements. These include three types of movements, right turn (R), left turn (L), and straight through (T), and can be applied to four different directions or bounds: eastbound (EB), westbound (WB), northbound (NB), and southbound (SB).

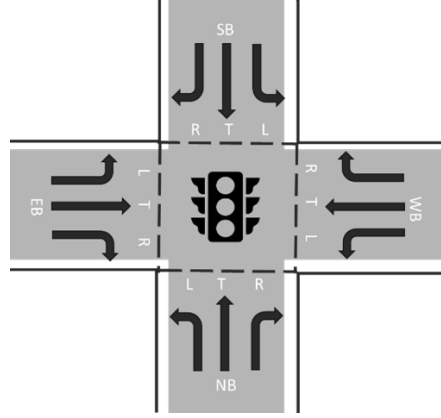


Figure 6.9: Traffic flow from different directions and turning movements

To solve the problem, the goal is to develop an accurate and reliable model that can be applied to perform real-time traffic flow forecasting for all turning movements at an intersection,

$$X_{M,(t+1,t+p)} = F(X_{M,(t-h,t)}) \quad (6.1)$$

The accuracy of the forecasting can be measured by the metrics that quantifies the closeness between $F(X_{M,(t-h,t)})$ and $X_{M,(t+1:t+p)}$.

Figure 6.3 illustrates the overarching framework proposed for developing and evaluating encoder-decoder and decoder-only models for few-shot and zero-shot scenarios. We divided our TMFs datasets into a training dataset and an evaluation dataset based on the city (Region of Waterloo or City of Milton) to which the intersection belongs. The training dataset is used to train the encoder-decoder models, while evaluation dataset is to assess the performance of both encoder-decoder and decoder-only models under few-shot and zero-shot scenarios. To enhance predictive accuracy, the TMFs dataset is first combined with four future-known exogenous covariate datasets including intersection and road layout data, urban planning data from municipal databases (Nie et al., 2022), and weather data from Environment Canada. We extracted three types of features from the raw datasets: TMFs time-domain features (historical temporal dependences), static features (intersection layout and location), and time-varying features (seasonality and weather). To reduce feature redundancy, we applied Spearman’s rank (Zar, 2005) correlation test to only keep the most relevant features. The retained features are then used to construct into training and evaluation datasets. The training dataset is utilized to train the encoder-decoder models, including TFT and DeepAR. Finally, both decoder-only and encoder-decoder models are evaluated on the evaluation dataset to facilitate a comprehensive performance comparison.

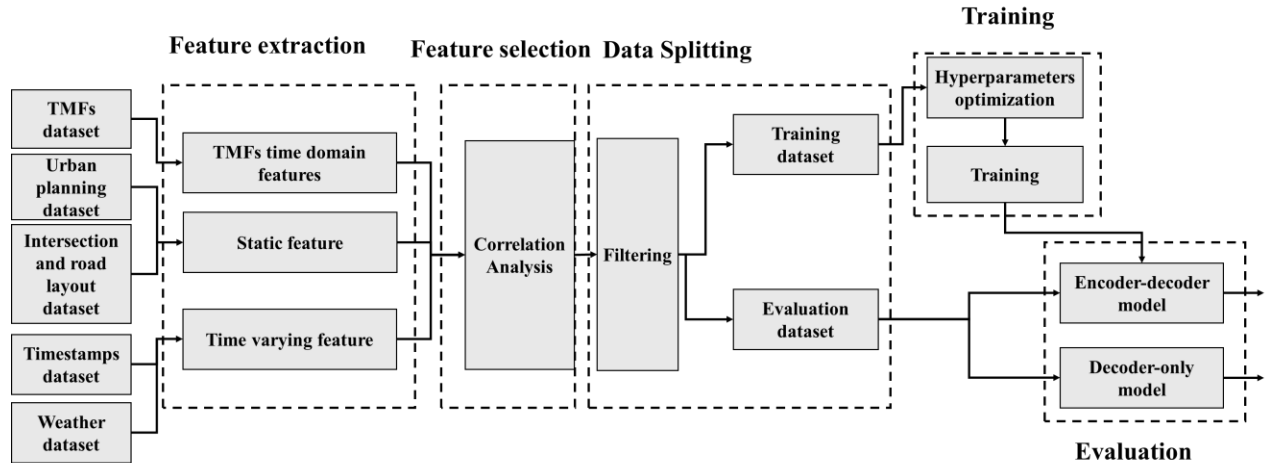


Figure 6.10: Research Methodology

6.2.1 TimesFM

As previously mentioned, we employ TimesFM as the primary decoder-only model for traffic turning movement forecasting in zero-shot and few-shot scenarios in this study. TimesFM is designed to generalize across unseen datasets using a patched-decoder architecture. During forecasting, raw TMFs time-series data is segmented into non-overlapping patches (Nie et al., 2022) to create structured input sequences, reducing the computational burden and improving efficiency. Then, each patch is encoded using a residual block with positional encodings, preserving both local short-term fluctuations and long-term dependencies. After that, the processed embeddings are passed through stacked multi-head self-attention layers, capturing temporal dependences in the data. Finally, TimesFM employs a hybrid decoding strategy, where predictions are generated in patches rather than one step at a time, striking a balance between auto-regressive generation and direct multi-step forecasting. This reduces error accumulation over longer horizons and enhances stability.

Despite its strengths, TimesFM has two primary limitations for traffic forecasting: Firstly, TimesFM lacks explicit mechanisms to learn trend, seasonality, and noise from training data, which can reduce forecasting accuracy in highly dynamic traffic conditions; Secondly, the original TimesFM is unable to incorporate external covariates (e.g., weather conditions, intersection layouts, and temporal factors), which are crucial in traffic flow modeling. To address this limitation, we enhanced TimesFM by integrating Empirical Mode Decomposition (EMD) (Rilling et al., 2003) and Random Forest (RF) (Rigatti, 2017):

- EMD as preprocessing step: EMD is employed as a preprocessing step to systematically decompose raw traffic flow data into Intrinsic Mode Functions (IMFs) and residuals before feeding them into TimesFM. This separation isolates dominant patterns, including long-term trends (captured in lower IMFs like IMF 5 and IMF 6), seasonal variations (reflected in medium-frequency IMFs like IMF 3 and IMF 4), and high-frequency fluctuations (represented in higher IMFs like IMF 1 and IMF 2) (Figure 4). The refined, noise-reduced signals are then fed individually into TimesFM, enhancing predictive accuracy and robustness by focusing on meaningful signals.

- RF as post-processing step: RF is trained on the residuals between TimesFM’s baseline predictions and actual observed traffic flows. The input to RF includes both the residuals and external covariates, such as weather data, intersection layout, and time-of-day variables, allowing the model to adjust TimesFM’s output based on real-time influences. This adjustment ensures that the model accounts for sudden traffic variations caused by external factors, improving predictive accuracy and robustness.

To evaluate the impact of these enhancements, we develop three TimesFM-based models (Figure 6.5): TimesFM, the original decoder-only model without modifications; TimesFM (EMD), a version that integrates EMD to improve signal extraction and reduce forecasting noise; and TimesFM (EMD&RF), a version that further incorporates RF to model external influences. By leveraging these enhancements, we expected TimesFM not only excels in zero-shot forecasting but also becomes more robust in capturing external influences and dynamic traffic variations. The next sections present the experimental results of these models in real-world TMFs prediction tasks.

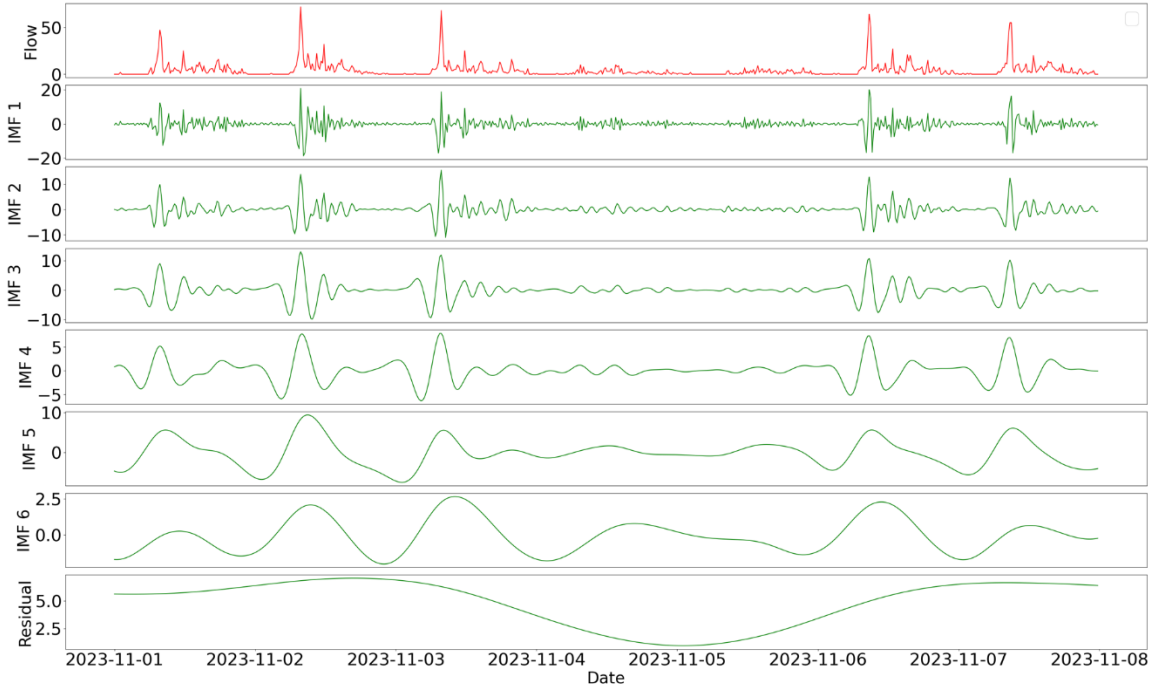


Figure 6.11: EMD example on TMFs dataset

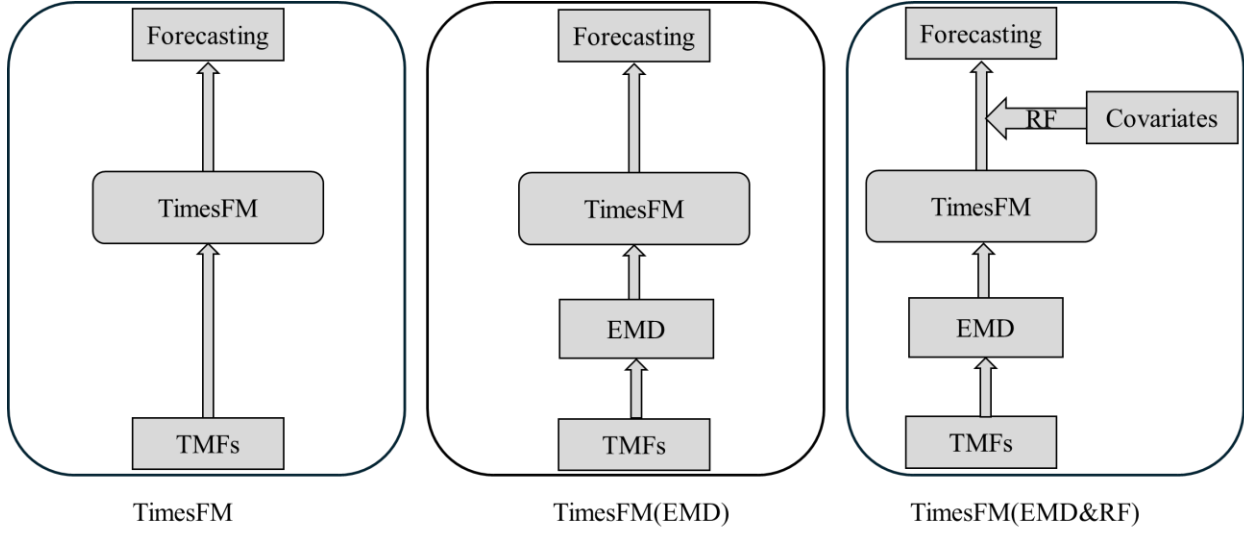


Figure 6.12: Three alternative TimesFM models

6.2.2 Encoder-decoder models for comparison

To evaluate the effectiveness of TimesFM in few-shot and zero-shot forecasting scenarios, we compare it against three established encoder-decoder models: Temporal Fusion Transformer (TFT)(Lim et al., 2021), DeepAR(Salinas et al., 2020), and Long Short-Term Memory (LSTM)(Sherstinsky, 2020).

TFT is an attention-based encoder-decoder model designed for multi-horizon time series forecasting. It applies multi-head self-attention mechanisms to identify relevant past time steps and integrates static and time-varying covariates to improve accuracy. TFT is particularly advantageous for structured datasets where external factors influence predictions. However, its performance depends on the availability of sufficient training data, making it less effective for zero-shot forecasting.

DeepAR is an encoder-decoder model based on autoregressive recurrent neural networks (RNNs). It utilizes Long Short-Term Memory (LSTM) or Gated Recurrent Unit (GRU) networks to model temporal dependencies and shared patterns across multiple related time series. DeepAR incorporates both dynamic and static covariates, enhancing forecasting accuracy and producing

probabilistic forecasts by estimating distributions over future values. Similar to TFT, its reliance on extensive historical data limits its generalization in zero-shot settings.

LSTM is a sequential learning model commonly used for time-series forecasting. It captures long-term dependencies through memory cells that retain relevant information while discarding irrelevant signals. LSTM is effective in structured environments but struggles with zero-shot forecasting, as it requires substantial historical data and does not incorporate external covariates as efficiently as attention-based models.

These encoder-decoder models serve as a benchmark for evaluating how well TimesFM generalizes across different traffic forecasting scenarios, particularly in cases with limited or no historical data.

6.3 Experimental results

All the encoder-decoder models, including TFT, DeepAR, and LSTM, were implemented with PyTorch (Paszke et al., 2019). To ensure model robustness, cross-validation was applied at each iteration, with 20% of the training dataset randomly selected for model validation. The model's hyperparameter was determined by the built-in Optuna (Liaw et al., 2018), an open-source hyperparameter optimization framework for searching optimal model parameters using Bayesian optimization. Based on the preliminary result, we determined that using 96 historical time steps (15 minutes per step, 24-hour in total) as input provided the most stable and accurate forecasting results. Consequently, this configuration was adopted for all model training and evaluation. The following section presents the dataset, evaluation metrics, and key findings from various computational experiments.

6.3.1 Case study data description

The Traffic Turning Movement Flow (TMFs) datasets were collected using Miovision's SmartSense system, a video-based detection system that continuously records turning movement counts 24/7. This study utilizes TMFs datasets from two independent jurisdictions: the Region of

Waterloo and the City of Milton, Ontario, Canada. The locations of these cities are shown in **Figure 6.6**.



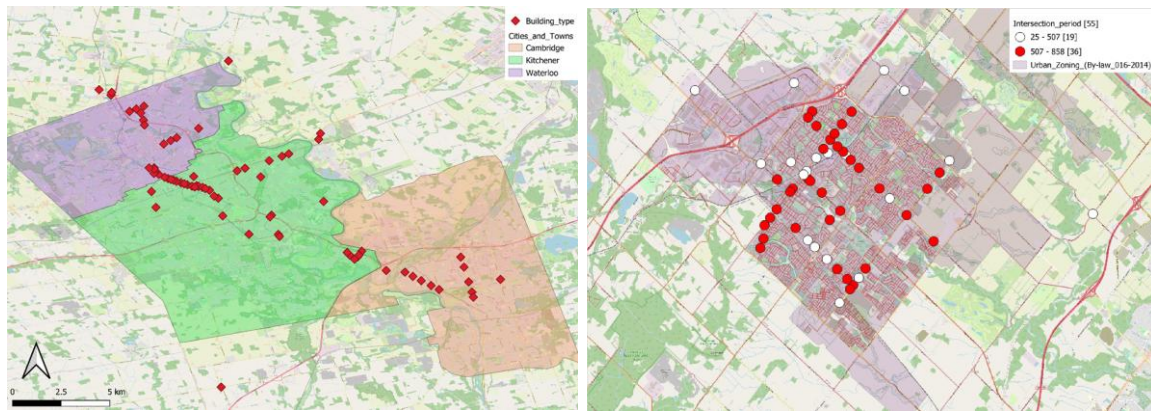
Figure 6.13: Case study sites

For training encoder-decoder models and conducting few-shot evaluations, we used the dataset from the Region of Waterloo, which includes data from 76 intersections (Figure 6.7(a)) over a 507-day period (January 1, 2022 – May 22, 2023). The data were aggregated into 15-minute intervals, resulting in 48,672 records per intersection and a total of 3,699,072 records. For training, data from 70 intersections (January 1, 2022 – March 22, 2023) were used, while the remaining six intersections were held out to evaluate the performance of both TimesFM and encoder-decoder models over the same period as few-shot scenarios.

For zero-shot evaluation, we used data from the City of Milton, covering 36 intersections (Figure 6.7(b)) from November 1, 2022 – February 27, 2024 (507 days). This dataset was also aggregated into 15-minute intervals, resulting in 48,672 records per intersection and a total of 1,752,192 records.

To enhance forecasting performance, we integrated additional datasets, including intersection and road layout data, urban planning data from municipal databases (Nie et al., 2022), and weather data from Environment Canada. These exogenous factors were integrated with the TMFs datasets, and Spearman’s rank correlation (Rasul et al., 2023) test was applied to remove

redundant variables. The selected inputs are summarized in Table 6.1. The selected exogenous features are divided into two types: prior-known time-varying covariates and static covariates. Static covariates are used as model inputs for training and evaluation of encoder-decoder models, Prior-known future time-dependent covariates are used for training encoder-decoder models. They are also used to evaluate both decoder-only and encoder-decoder models under few-shot and zero-shot scenarios.



a. Region of Waterloo

b. City of Milton

Figure 6.14: Intrumented intersections at the two locations

Table 6.4: External covariates

Variable types	Variables	Descriptions
Prior-Known Future Time-Dependent Covariates	Weekday	Day of the week
	Hour	Hour of the day (24-hour format)
	Year	Year
	Month	Month (e.g., January, February, etc.)
	Holiday	Public holiday in Ontario
	Visibility	Road visibility (100m)
	Weather_eng_desc	Weather categories (rain, snow, etc.)
	Precip_amount	Amount of precipitation(cm)
	Temp	Temperature (°C)
Static Covariates	Cartoclass	Road category (Arterial, Collector, Local Street, Ramp)
	Flowdirection	Direction (Northbound, Southbound, Eastbound, Westbound)
	Speedzone	Posted speed limit
	Dividedroad	Indicates if the road is divided or not
	Numberoflanes	Number of lanes on the road
	Landuse	Land use category (Assembly, Business, Educational, Industrial, Institutional, Mercantile, Residential, missed)

6.3.2 Performance measures

Several performance measures have been utilized to evaluate the effectiveness of the TMFs forecasting models. In this study, five widely used metrics are applied: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Root Mean Squared Error (RMSE), Root Mean Squared Percentage Error (RMSPE), and Variance of Absolute Percentage Error (VAPE). MAE calculates the average absolute differences between predicted and actual values, while RMSE measures the square root of the average squared differences, providing a sense of the error's magnitude. Notably, MAE treats all errors equally, whereas RMSE gives more weight to

larger errors, making it sensitive to outliers. RMSPE, derived as percentage-based counterparts of RMSE, are scale-independent metrics, enabling comparisons across datasets with varying scales. This makes them particularly useful for evaluating forecasting performance in diverse traffic flow scenarios. VAPE further quantifies the variability in percentage errors, providing additional insights into model consistency. The equations for MAE, MAPE, RMSE, and RMSPE are defined as follows.

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (6.2)$$

$$VAPE = Var\left(\frac{y_i - \hat{y}_i}{y_i}\right) \times 100 \quad (6.3)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (6.4)$$

$$RMSPE = 100\% \times \sqrt{\frac{1}{n} \sum_{i=1}^n \left(\frac{y_i - \hat{y}_i}{y_i}\right)^2} \quad (6.5)$$

Where y_i is observation TMFs at the i th time interval, $\hat{y}_i$ is the predicted TMFs at the i th time interval, and n is the number of the observation.

To further evaluate model performance, we apply Dynamic Time Warping (DTW) cumulative distance (Müller, 2007) to quantify the temporal misalignment between predicted and observed TMFs. DTW is a time-series alignment method that measures similarity by allowing non-linear warping of the time axis. It computes the optimal alignment path between two sequences by minimizing the cumulative Euclidean distance while preserving temporal constraints. This makes DTW particularly useful for traffic forecasting, where the timing of predicted and observed turning movement flows may not align perfectly. By analyzing the alignment path, DTW cumulative distance provides valuable insights into forecasting lag, determining how much predictions systematically lead or lag behind real-world observations. This metric is particularly useful for evaluating how well a forecasting model adapts to sudden changes in traffic flow patterns.

Euclidean Distance:

$$d(\mathbf{y}_t, \hat{\mathbf{y}}_t) = \sqrt{\sum_1^n (\mathbf{y}_t - \hat{\mathbf{y}}_t)^2} \quad (6.6)$$

Cumulative DTW Euclidean Distance by

$$c(\mathbf{y}_t, \hat{\mathbf{y}}_t) = d(\mathbf{y}_t, \hat{\mathbf{y}}_t) + \min \begin{cases} c(\mathbf{y}_{t-1}, \hat{\mathbf{y}}_t) \\ c(\mathbf{y}_t, \widehat{\mathbf{y}}_{t-1}) \\ c(\mathbf{y}_{t-1}, \widehat{\mathbf{y}}_{t-1}) \end{cases} \quad (6.7)$$

Where $\mathbf{y}_t$ is the observation TMFs at the t th time interval, $\hat{\mathbf{y}}_t$ is the predicted TMFs at the t th time and n is the number of the observation. And the initial dot is $c(\mathbf{y}_{t0}, \widehat{\mathbf{y}}_{t0}) = d(\mathbf{y}_{t0}, \widehat{\mathbf{y}}_{t0}) = 1$.

6.3.3 Model performance for different datasets

We conducted the first experiment for one-step (15 minutes)-ahead forecasting in three different scenarios: (1) test scenario, which evaluates model performance on a held-out portion from the same distribution as the training data; (2) few-shot scenario, where models are tested on intersections not seen during training but from the same city (Region of Waterloo); and (3) zero-shot scenario, assessing model generalization to a different city with entirely unseen intersections. The performance metrics for each setup are summarized in the following tables.

Table 6.5: Comparison of forecasting models on test dataset

	Test dataset					
	TimesFM	TimesFM(EMD)	TimesFM(EMD&RF)	TFT	DeepAR	LSTM
RMSE (vehs/h)	15.46	12.92	11.68	7.10	10.89	15.73
RMSPE (%)	25.72	14.12	13.70	10.48	17.18	24.43
MAE (vehs/h)	24.82	19.10	18.28	13.27	17.17	22.90
VAPE (%)	0.42	0.33	0.33	0.36	0.41	0.60

Table 6.6: Comparison of forecasting models on hold-out dataset (Few-shot scenarios)

	Hold Out					
	TimesFM	TimesFM(EMD)	TimesFM(EMD&RF)	TFT	DeepAR	LSTM
RMSE (vehs/h)	19.13	10.72	10.69	7.73	10.82	15.29
RMSPE (%)	20.29	19.20	17.64	18.38	19.67	23.40
MAE (vehs/h)	12.15	6.50	6.27	5.51	7.45	10.37
VAPE (%)	0.52	0.44	0.38	0.33	0.54	0.79

Table 6.7: Comparison of forecasting models on zero-shot scenarios

	Zero Shot					
	TimesFM	TimesFM(EMD)	TimesFM(EMD&RF)	TFT	DeepAR	LSTM
RMSE (vehs/h)	16.55	14.87	9.64	11.52	16.53	18.69
RMSPE (%)	24.42	16.23	8.12	12.92	22.11	38.51
MAE (vehs/h)	16.92	9.05	7.32	8.17	20.93	21.83
VAPE (%)	0.56	0.40	0.33	0.36	0.43	0.83

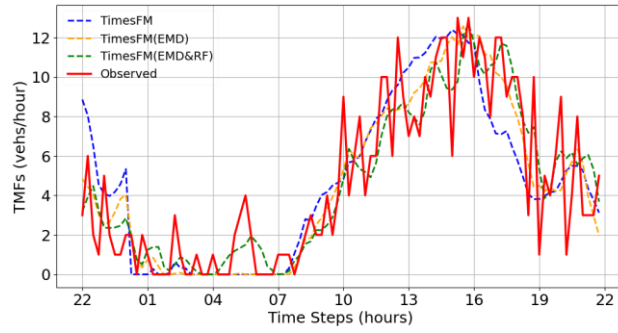
For the test dataset, TimesFM(EMD&RF) outperforms all other decoder-only models, suggesting that the integration of EMD and RF significantly enhances TimesFM’s predictive accuracy. TFT achieves the best overall performance, demonstrating that attention-based models are highly effective in structured training environments. DeepAR ranks second, while LSTM shows the weakest performance. Additionally, TimesFM(EMD) surpasses the base TimesFM, reinforcing the benefits of decomposition techniques for noise reduction and improved pattern extraction.

For the hold-out dataset, TFT again achieves the lowest RMSE, MAE, and VAPE, indicating that attention-based models generalize well within similar environments. TimesFM (EMD&RF) remains competitive, while TimesFM (EMD) continues to outperform the base TimesFM, confirming that EMD enhances predictive accuracy by filtering out irrelevant fluctuations. The baseline TimesFM and LSTM remain the weakest performers, suggesting unstable predictions and high variance.

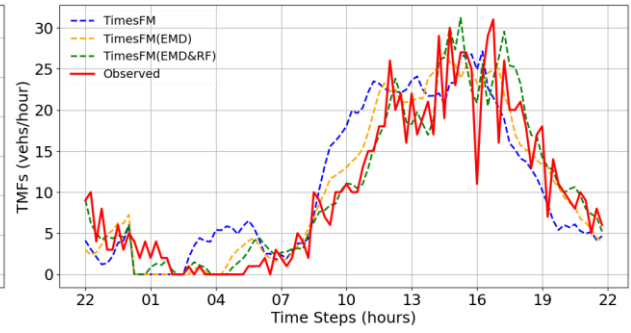
In the zero-shot scenario, where models forecast TMFs for an unseen city, TimesFM(EMD&RF) outperforms all models, achieving the lowest RMSE, RMSPE, MAE, and VAPE, confirming that the integration of EMD and RF enables TimesFM to adapt effectively to new locations without retraining. TFT performs reasonably well but slightly trails TimesFM(EMD&RF), indicating that attention-based models require some domain-specific tuning to reach optimal performance. TimesFM(EMD) again shows notable improvements over the base TimesFM, reinforcing the value of decomposition in handling new traffic conditions. In contrast, LSTM remains the weakest performer, highlighting its limitations in generalizing to unseen datasets due to its reliance on long-term sequential dependencies.

These findings confirm that TFT is the most robust model in test and hold-out scenarios, while TimesFM(EMD&RF) excels in zero-shot scenarios, demonstrating that hybrid approaches integrating decomposition techniques and external covariates can significantly improve forecasting accuracy. DeepAR also performs well in structured environments but shows slight limitations in zero-shot settings, while LSTM consistently underperforms across all datasets, reinforcing the need for advanced modeling techniques to improve generalization in TMFs forecasting.

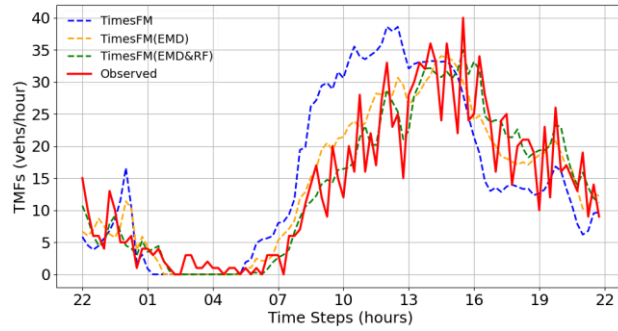
Figure 6.8 shows the forecasted TMFs from three decoder-only models on a sample zero-shot scenario. The experimental setup for evaluating TMFs forecasting involves comparing the predicted and observed traffic flow patterns over a 24-hour period across multiple turning movements at an example intersection in the City of Milton.



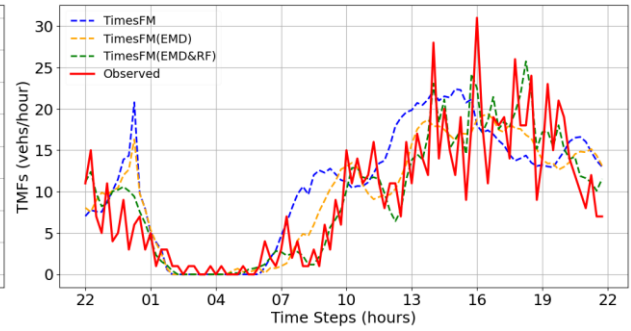
(a) Eastbound left turn



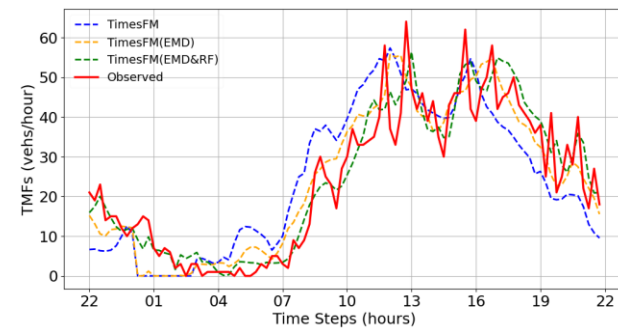
(b) Eastbound right turn



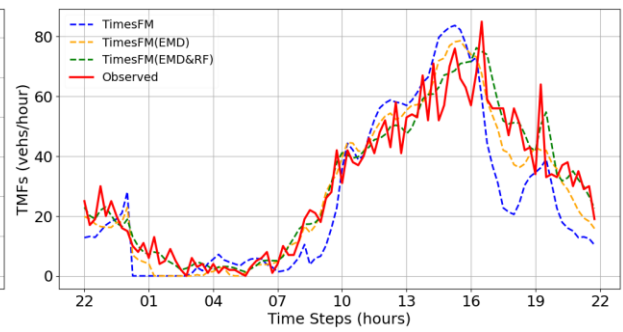
(c) Eastbound straight through



(d) Northbound left turn



(e) Northbound right turn

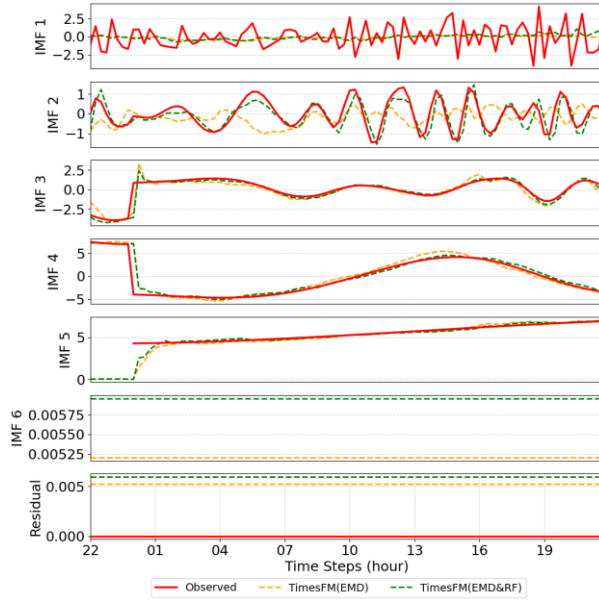


(f) Northbound straight through

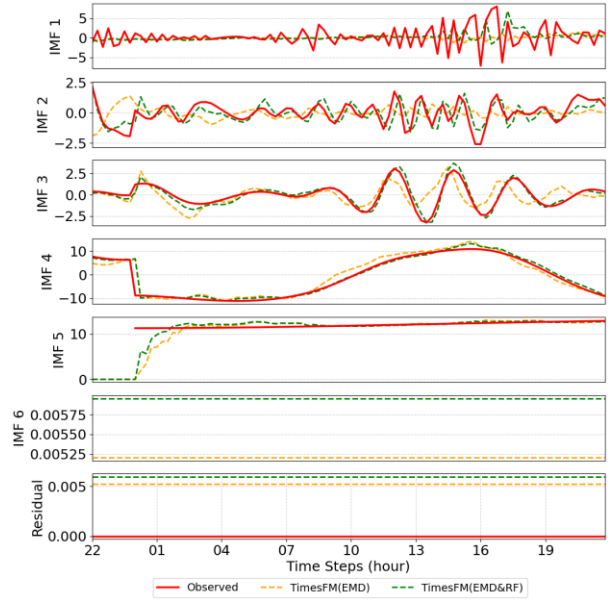
Figure 6.15: Sample of observed and predicted TMFs.

The forecasting results for different TMFs over a 24-hour period show that TimesFM(EMD&RF) consistently provides the best alignment with observed traffic flow,

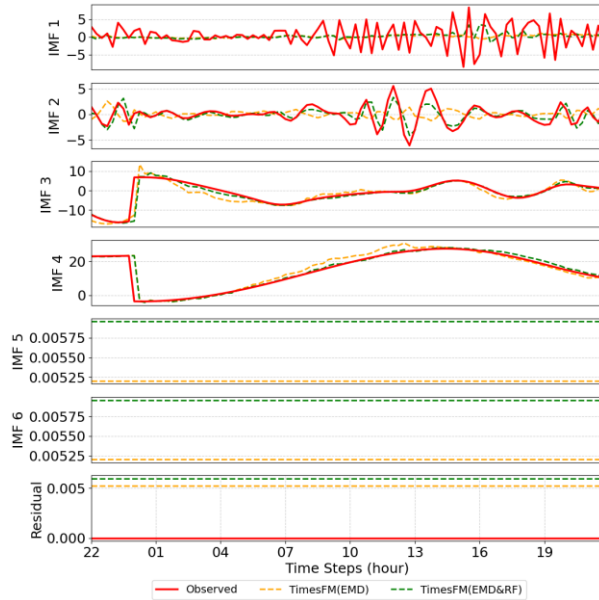
accurately capturing both peak and off-peak variations due to its integration of EMD preprocessing and exogenous covariates. TimesFM(EMD) also improves over the original TimesFM, demonstrating that EMD effectively reduces noise and enhances local pattern matching.



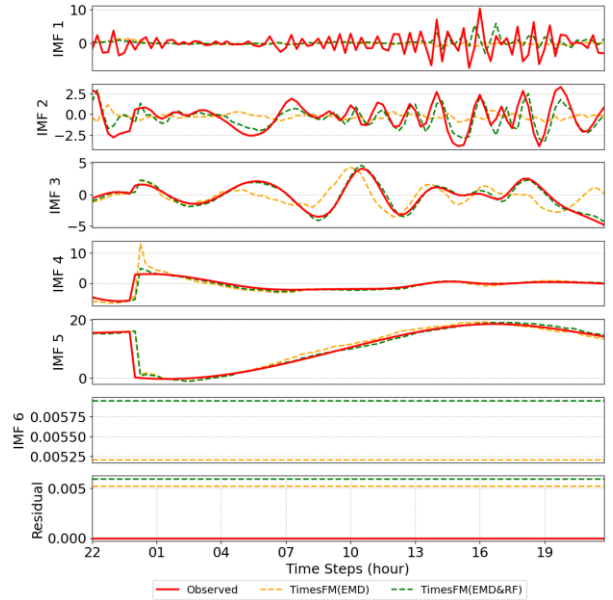
(a) Eastbound left turn



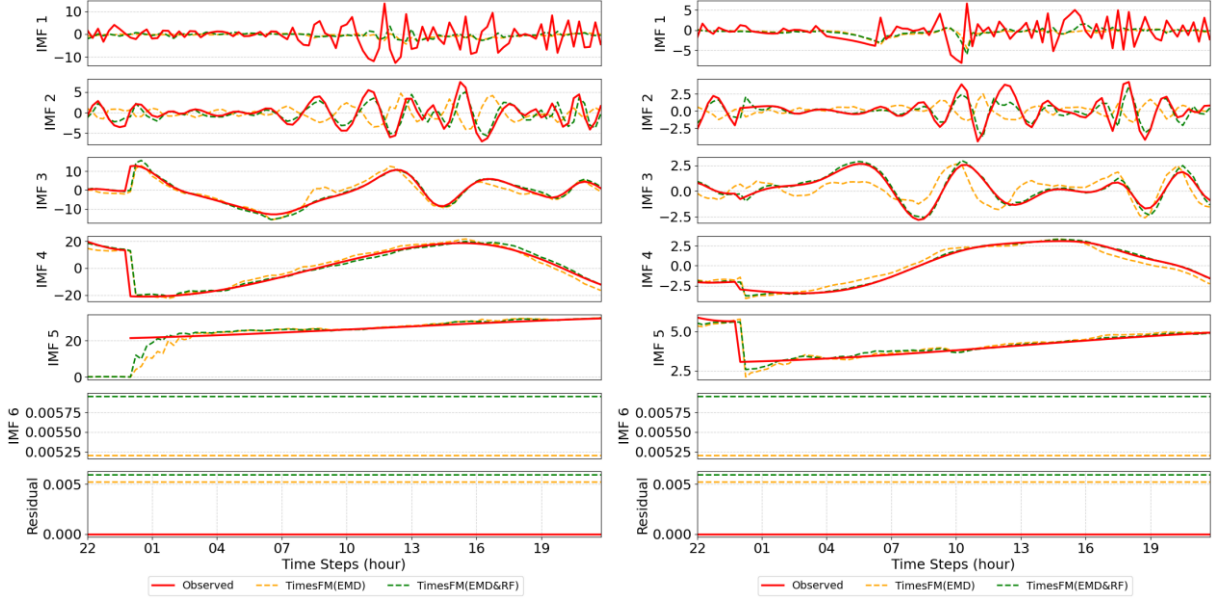
(b) Eastbound right turn



(c) Eastbound straight through



(d) Northbound left turn



(e) Northbound right turn

(f) Northbound straight through

Figure 6.16: TimesFM Forecasting EMD decomposition examples on Dec 2nd, 2023, at intersection Fairway at Wilson.

Figure 6.9 shows the decomposition of TMFs forecasting results by TimesFM(EMD) and TimesFM(EMD&RF) into IMFs using EMD. This highlights the distinct frequency components that are present within the different traffic flow dynamics shown in Figure 6.8. High-frequency IMFs (top panels) represent short-term fluctuations influenced by signal timing and sudden congestion, where TimesFM(EMD&RF) (green, dash-dot line) follows the observed fluctuations best. Mid-frequency IMFs (middle panels) capture daily traffic patterns, such as morning and evening peak congestion, and both TimesFM(EMD&RF) and TimesFM(EMD) (orange, dashed line) closely follow the observed values, demonstrating the effectiveness of EMD in enhancing prediction stability. Low-frequency IMFs (bottom panels) represent long-term trends, showing consistent alignment across all models, suggesting that global traffic patterns are easier to capture than short-term variations. Overall, TimesFM(EMD&RF) outperforms other models across all IMF levels, confirming that combining RF with EMD improves accuracy in capturing both short-term fluctuations and long-term traffic trends.

6.3.4 Multistep forecasting comparison

For this experiment, all models are tested at increasing forecast horizons, at 1-step, 4-step, 24-step, and 96-step (or 15 min, 1 hour, 6 hours, and 24 hours), to analyze how errors propagate over time in zero-shot scenarios.

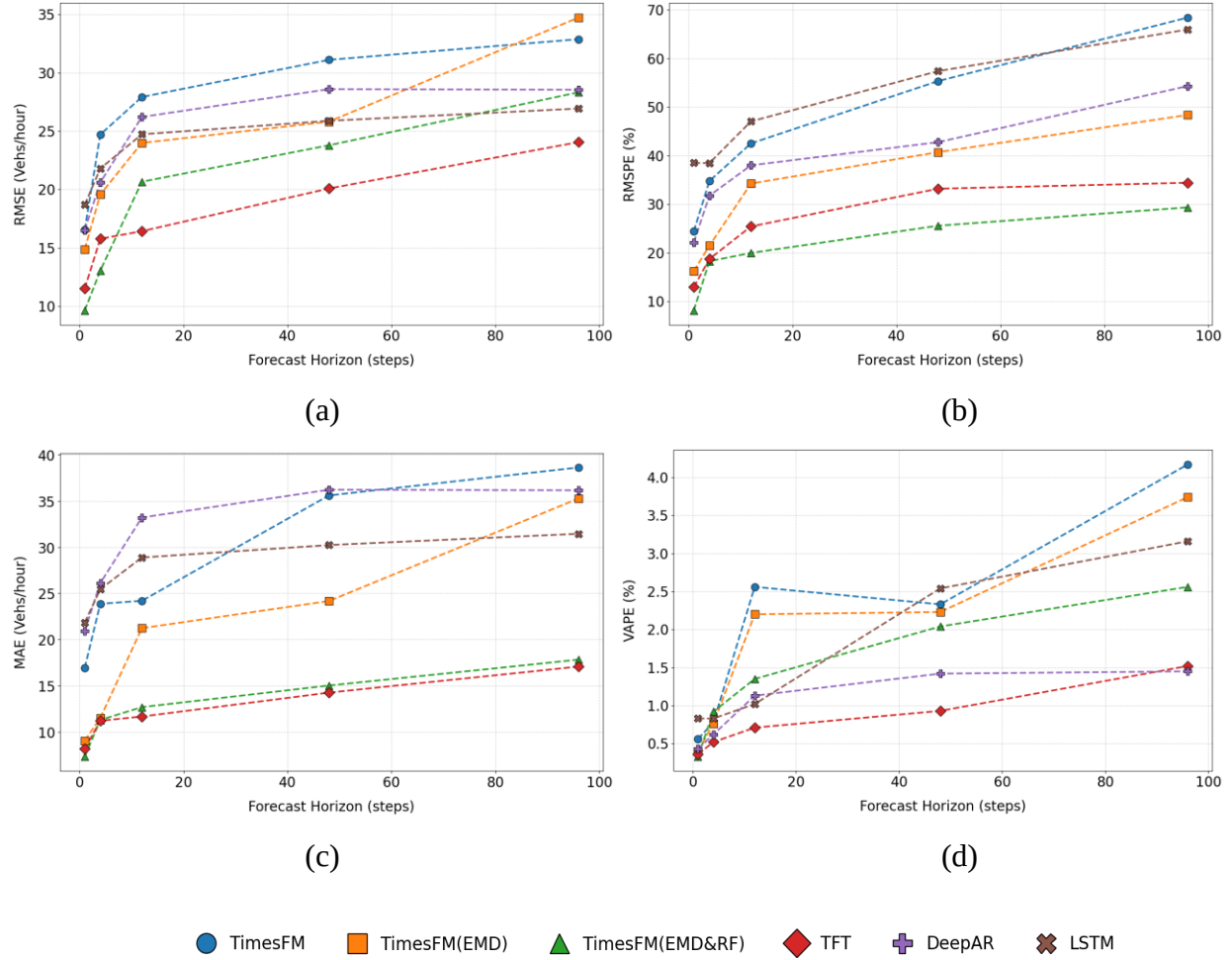


Figure 6.17: RMSE, RMSPE, MAE and VAPE for different forecasting horizons.

The results show that TimesFM(EMD&RF) outperforms all other models at short horizons in terms of RMSE, RMSPE, MAE, and VAPE, highlighting the effectiveness of combining EMD decomposition and exogenous covariates in enhancing short-term forecasting accuracy and stability. However, its performance degrades as the forecasting horizon increases, suggesting

greater sensitivity to uncertainty accumulation, particularly in terms of RMSE and VAPE. TFT achieves the best performance in long-term forecasting, especially beyond 24 steps (six hours), demonstrating its strength in handling multi-horizon forecasts. TimesFM(EMD), which incorporates EMD preprocessing alone, also shows notable improvements over the base TimesFM, which consistently performs the worst, indicating the benefit of decomposing traffic flow into intrinsic patterns to reduce noise. Overall, the results demonstrate that decoder-only models like TimesFM(EMD&RF) provide strong short-term forecasting performance but are less effective than TFT for long-term multi-horizon TMFs forecasting. Compared with other supervised models under special events

In order to assess the temporal alignment between predicted and observed TMFs for regular peaks and special events, this experiment applies the Dynamic Time Warping (DTW) distance metric on an hourly basis to evaluate model performance under different weather conditions, including Snow, Rain, Cloudy, and Clear. To avoid excessively large DTW values that could obscure meaningful patterns, 100 hours were randomly sampled for each weather category. The DTW distance per hour is calculated to quantify the degree of misalignment and lag between the predicted and observed traffic flows for one-step-ahead forecasting. A lower DTW distance indicates better alignment and faster responsiveness of the model to traffic changes, which is particularly crucial in adverse weather scenarios where accurate and timely forecasts are critical for traffic management.

Table 6.8: DTW distance for different models for different weathers

DTW distance per hour						
Weather	TimesFM	TimesFM(EMD)	TimesFM(EMD&RF)	TFT	DeepAR	LSTM
Snow	16.87	12.26	8.41	9.24	11.05	23.35
Rain	21.32	16.13	11.99	11.78	13.53	29.86
Cloudy	27.03	18.28	12.81	12.44	14.64	29.58
Clear	28.05	17.37	12.22	12.56	14.62	26.85

The Table 6.5 presents the DTW distance per hour for different forecasting models under varying weather conditions for the zero-shot scenario. From the table, it can be found that

TimesFM(EMD&RF) consistently achieves the lowest DTW distance across all weather conditions, with values of 8.41 (Snow), 11.99 (Rain), 12.81 (Cloudy), and 12.22 (Clear), indicating that combining EMD and exogenous covariates effectively reduces the temporal misalignment and improves responsiveness to traffic fluctuations across different weather scenarios. TimesFM(EMD), which applies EMD preprocessing without additional exogenous modeling, also outperforms the baseline TimesFM, reducing DTW distance in all weather conditions. TFT achieves competitive performance, with DTW values close to TimesFM(EMD&RF) under normal weather conditions (e.g., Cloudy and Clear), highlighting the robustness of attention-based architectures in capturing complex patterns during stable weather. DeepAR performs moderately, while LSTM demonstrates the highest DTW distances in all conditions, particularly under Snow (23.35) and Rain (29.86), indicating its struggle to adapt to sudden traffic changes induced by adverse weather. These results emphasize that integrating decomposition methods and external factors improves a model’s ability to capture the dynamic nature of traffic flows, particularly under varying weather conditions.

6.4 Conclusions

Accurately forecasting Traffic Turning Movement Flows (TMFs) is essential for effective urban traffic management; however, traditional supervised models, such as TFT, DeepAR, and LSTM (evaluated in earlier chapters), struggle in zero-shot and few-shot scenarios due to their heavy reliance on extensive historical data. This study introduces and evaluates decoder-only models—specifically TimesFM—to address these limitations. By integrating Empirical Mode Decomposition (EMD) for advanced feature extraction and Random Forest (RF) to incorporate critical external covariates, TimesFM’s adaptability and forecasting accuracy in dynamic urban environments are substantially improved.

A comparative analysis demonstrates that the enhanced TimesFM (EMD&RF) significantly surpasses the performance of previously discussed models, including TFT and DeepAR, particularly in zero-shot forecasting scenarios where minimal historical data is available. While the TimesFM(EMD&RF) model maintains competitive performance in few-shot and standard forecasting tasks, DTW analysis further highlights its strength by showcasing substantial

reductions in prediction lag, contributing to more responsive and reliable forecasting outcomes. Additionally, the evaluation under diverse weather conditions emphasizes the critical role of external factors in TMFs forecasting, reinforcing the importance of hybrid modeling approaches that effectively integrate exogenous variables.

Nevertheless, despite their strong generalization capabilities, decoder-only models such as TimesFM still face challenges. They do not explicitly account for scheduled events like road closures or non-recurrent disruptions—events that significantly impact real-world traffic patterns. Moreover, further improving the ability of these models to capture spatial dependencies and enhancing computational efficiency remain essential for practical applicability and large-scale deployment.

Chapter 7: Conclusions

7.1 Summary of major studies and findings

This research developed and systematically investigated advanced machine learning approaches for short-term intersection turning movement flows (TMFs) forecasting, focusing on model performance, computational efficiency, and real-world applicability. The key findings across the studies are summarised as follows:

In Chapter 3, PB-LSTM, a novel stacked bidirectional LSTM (SBiLSTM) combined with a multilayer perceptron (MLP) was developed to capture multi-scale temporal patterns in TMFs. The PB-LSTM demonstrated consistent performance advantages over baseline models such as SBiLSTM and MLP, particularly when trained on shorter input sequences (30 minutes vs 60 minutes). Notably, it also exhibited lower sensitivity of performance degradation to forecasting horizons compared to other models, showing its robustness for real-time applications. While intersection clustering strategies were introduced to enhance generalizability across hold-out intersections, the observed improvements remained moderate, suggesting the need for refined clustering methodologies in future work.

In Chapter 4, the DeepAR framework was applied to generate probabilistic TMFs predictions with uncertainty quantification. Results showed that DeepAR outperformed Transformer and ARIMA models in multi-step forecasting accuracy while producing reliable confidence intervals, a critical feature for risk-aware traffic management decisions. The inclusion of time-dependent covariates, such as temporal features and environmental factors, significantly reduced prediction uncertainty. Furthermore, fine-tuning the model on hold-out intersections with limited training data confirmed its strong transferability, underscoring its practical utility in scenarios with sparse historical records.

In Chapter 5, a Temporal Fusion Transformer (TFT)-based pipeline was proposed to integrate historical TMFs data, static covariates, and future time-varying inputs (e.g., intersection geometry, weather conditions). Feature importance analysis identified speed zone and road

category as the most influential static covariates, while hour of day and temperature emerged as dominant time-dependent factors. Despite its ability to maintain consistent performance across diverse traffic patterns, the TFT model struggled to predict extreme traffic volumes, revealing a critical limitation that necessitates adaptive mechanisms such as quantile loss integration or specialized peak-hour modeling.

In Chapter 6, the decoder-only TimesFM model, enhanced with Empirical Mode Decomposition (EMD) for feature extraction and Random Forest (RF) for covariate integration, was evaluated for zero-shot and few-shot forecasting. In zero-shot scenarios, TimesFM (EMD&RF) achieved competitive performance against encoder-decoder architectures like TFT and DeepAR, with DTW analysis confirming reduced prediction lag and improved responsiveness. The model also demonstrated robustness under diverse weather conditions, though its performance in zero-shot scenarios highlighted the persistent influence of external factors such as sudden weather changes.

7.2 Summary of major contributions

This research presents a systematic effort to improve real-time intersection Turning Movement Flow (TMFs) forecasting by applying state-of-the-art deep learning techniques and introducing innovative methodologies. The key contributions of this research are:

1. Advanced models for capturing complex spatial and temporal patterns

This research developed a novel Parallel Bidirectional LSTM (PB-LSTM) architecture by uniquely integrating stacked bidirectional LSTM layers with a traditional multilayer perceptron (MLP). This innovative design explicitly addresses the limitations of existing models by using MLP to effectively capture seasonal patterns and LSTM layers to model short-term temporal dependencies in TMFs data, significantly improving forecasting accuracy and reliability.

2. Probabilistic framework for explicit uncertainty quantification

This research developed a practical probabilistic forecasting approach based on the DeepAR model, explicitly quantifying uncertainty in TMFs predictions. By integrating important external variables—such as weather conditions, road type, and time of day—this research substantially

reduced prediction uncertainty and produced meaningful confidence intervals, thereby enhancing the decision-making capabilities of urban traffic management systems.

3. Detailed analysis and integration of external factors

This research designed a forecasting pipeline based on the Temporal Fusion Transformer (TFT) to effectively incorporate and analyze a broad range of exogenous factors, including road characteristics, speed zones, weather, and time-related features. Through rigorous feature importance analysis, this research identified and quantified how these factors impact prediction accuracy, thus improving model interpretability and performance in real-world traffic conditions. This has the potential to enable targeted traffic management and control schemes such as variable speed limits and dynamic lane management.

4. Robust and adaptive framework for zero-shot and few-shot forecasting

This research addressed the critical challenge of data scarcity by developing an adaptive forecasting framework. By innovatively combining EMD for pre-processing and RF for post-training fine-tuning, this research significantly enhanced the TimesFM model's capability to generalize effectively to previously unseen intersections without requiring extensive retraining. This approach improved the robustness of the TimesFM model for traffic forecasting across various forecasting horizons and under challenging weather conditions, ensuring consistent performance even at new intersections or cities.

5. Comprehensive evaluation and benchmarking across diverse scenarios

This research established a structured benchmarking methodology specifically designed to rigorously compare deep learning models—including PB-LSTM, TFT, DeepAR, and TimesFM—under various realistic forecasting conditions. This comprehensive analysis effectively highlighted the distinct strengths and weaknesses of each model, providing practical insights that can guide future model selection, optimization, and deployment. The findings are expected to support effective model deployment strategies, helping cities optimize resource allocation and operational efficiency.

Overall, this research underscores the necessity of hybrid approaches that balance intrinsic traffic patterns with external covariates for reliable real-world deployment. By addressing temporal

dependencies, uncertainty quantification, exogenous factor integration, and data scarcity challenges, this research advances the development of adaptive and interpretable frameworks for urban traffic management systems.

7.3 Recommendations for future research

Although this research provides significant advancements in intersection TMFs forecasting, several important areas remain to be explored in future studies:

1. Integration of additional data sources: Future studies could explore the integration of real-time data from connected vehicles, traffic cameras, and social media to enhance forecasting accuracy and adaptability.
2. Development of lightweight models for edge computing: Designing efficient models that can run on edge devices, such as traffic signal controllers, could enable real-time decision-making at the intersection level.
3. Investigation of spatiotemporal dependencies in complex networks: Exploring Graph Neural Networks (GNN) and Spatiotemporal Graph Convolutional Networks (ST-GCN) for capturing complex spatial relationships across multiple intersections could improve forecasting accuracy.
4. Adapting models to dynamic traffic environments: Future research could focus on developing adaptive learning models capable of real-time retraining and adjustment to account for sudden traffic pattern changes caused by incidents, roadwork, or events.
5. Extending model applicability to multimodal transportation systems: Investigating these models in mixed traffic environments—including transit, cycling, and pedestrians—and their integration with traffic signal optimization and traveler information systems could enhance urban mobility and congestion management.

References

- Agarap, A. F. (2018). Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*.
- Alghamdi, T., Elgazzar, K., Bayoumi, M., Sharaf, T., & Shah, S. (2019). Forecasting traffic congestion using ARIMA modeling. 2019 15th international wireless communications & mobile computing conference (IWCMC),
- Althelaya, K. A., El-Alfy, E.-S. M., & Mohammed, S. (2018). Evaluation of bidirectional LSTM for short-and long-term stock market prediction. 2018 9th international conference on information and communication systems (ICICS),
- Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.
- Biau, G., & Scornet, E. (2016). A random forest guided tour. *Test*, 25(2), 197-227.
- Cai, L., Janowicz, K., Mai, G., Yan, B., & Zhu, R. (2020). Traffic transformer: Capturing the continuity and periodicity of time series for traffic forecasting. *Transactions in GIS*, 24(3), 736-755.
- Castro-Neto, M., Jeong, Y.-S., Jeong, M.-K., & Han, L. D. (2009). Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions. *Expert systems with applications*, 36(3), 6164-6173.
- Chakrabarti, A., Sadhu, P. K., & Pal, P. (2022). AWS IoT Core and Amazon DeepAR based predictive real-time monitoring framework for industrial induction heating systems. *Microsystem Technologies*, 1-16.
- Chen, A., Chootinan, P., Ryu, S., Lee, M., & Recker, W. (2012). An intersection turning movement estimation procedure based on path flow estimator. *Journal of Advanced Transportation*, 46(2), 161-176.
- Chen, Y., Shu, L., & Wang, L. (2017). Traffic flow prediction with big data: A deep learning based time series model. 2017 IEEE conference on computer communications workshops (INFOCOM WKSHPS),

- Chikkakrishna, N. K., Hardik, C., Deepika, K., & Sparsha, N. (2019). Short-term traffic prediction using sarima and FbPROPHET. 2019 IEEE 16th India Council International Conference (INDICON),
- ChikkaKrishna, N. K., Rachakonda, P., & Tallam, T. (2022). Short-term traffic prediction using Fb-PROPHET and neural-PROPHET. 2022 IEEE Delhi Section Conference (DELCON),
- Clevert, D.-A., Unterthiner, T., & Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*.
- Consoli, S., Negri, M., Tebbifakhr, A., Tosetti, E., & Turchi, M. (2022). On neural forecasting and news emotions: the case of the Spanish stock market. Machine Learning and Principles and Practice of Knowledge Discovery in Databases: International Workshops of ECML PKDD 2021, Virtual Event, September 13-17, 2021, Proceedings, Part II,
- Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3), 273-297.
- Cui, Z., Henrickson, K., Ke, R., & Wang, Y. (2019). Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 21(11), 4883-4894.
- Cui, Z., Ke, R., Pu, Z., & Wang, Y. (2020). Stacked bidirectional and unidirectional LSTM recurrent neural network for forecasting network-wide traffic state with missing values. *Transportation Research Part C: Emerging Technologies*, 118, 102674.
- Daganzo, C. F. (1997). *Fundamentals of transportation and traffic operations*.
- Dahlgren, J., Weissenberger, S., Hickman, M., & Lo, H. (1996). Lessons from Case Studies of Advanced Transportation and Information Systems.
- Dai, H., Yang, Z., & Guo, S. (2006). Real-time traffic volume estimation with fuzzy linear regression. 2006 6th World Congress on Intelligent Control and Automation,
- Das, A., Kong, W., Sen, R., & Zhou, Y. (2023). A decoder-only foundation model for time-series forecasting. *arXiv preprint arXiv:2310.10688*.
- Dauphin, Y. N., Fan, A., Auli, M., & Grangier, D. (2017). Language modeling with gated convolutional networks. International conference on machine learning,
- De Vlieger, I., De Keukeleere, D., & Kretzschmar, J. (2000). Environmental effects of driving behaviour and congestion related to passenger cars. *Atmospheric Environment*, 34(27), 4649-4655.

- Deshpande, M., & Bajaj, P. R. (2016). Performance analysis of support vector machine for traffic flow prediction. 2016 international conference on global trends in signal processing, information computing and communication (ICGTSPICC),
- Dong, H., Jia, L., Sun, X., Li, C., & Qin, Y. (2009). Road traffic flow prediction with a time-oriented ARIMA model. 2009 Fifth International Joint Conference on INC, IMS and IDC,
- Du, S., Li, T., Yang, Y., & Horng, S.-J. (2020). Multivariate time series forecasting via attention-based encoder–decoder framework. *Neurocomputing*, 388, 269-279.
- Edwards, D. R., Handsley, M. M., & Pennington, C. J. (2008). The ADAM metalloproteinases. *Molecular aspects of medicine*, 29(5), 258-289.
- Ekambaram, V., Jati, A., Nguyen, N. H., Dayama, P., Reddy, C., Gifford, W. M., & Kalagnanam, J. (2024). TTMs: Fast Multi-level Tiny Time Mixers for Improved Zero-shot and Few-shot Forecasting of Multivariate Time Series. *arXiv preprint arXiv:2401.03955*.
- Ermagun, A., & Levinson, D. (2018). Spatiotemporal traffic forecasting: review and proposed directions. *Transport Reviews*, 38(6), 786-814.
- Falcon, W. A. (2019). Pytorch lightning. *GitHub*, 3.
- Farge, M. (1992). Wavelet transforms and their applications to turbulence. *Annual review of fluid mechanics*, 24(1), 395-458.
- FHWA. (2022). *TRAFFIC VOLUME TRENDS*.
- Fix, E., & Hodges, J. L. (1989). Discriminatory analysis. Nonparametric discrimination: Consistency properties. *International Statistical Review/Revue Internationale de Statistique*, 57(3), 238-247.
- Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1), 119-139.
- Fu, R., Zhang, Z., & Li, L. (2016). Using LSTM and GRU neural network methods for traffic flow prediction. 2016 31st Youth academic annual conference of Chinese association of automation (YAC),
- Fu, Z., Lam, W., Yu, Q., So, A. M.-C., Hu, S., Liu, Z., & Collier, N. (2023). Decoder-only or encoder-decoder? interpreting language model as a regularized encoder-decoder. *arXiv preprint arXiv:2304.04052*.

- Ghanim, M. S., & Shaaban, K. (2018). Estimating turning movements at signalized intersections using artificial neural networks. *IEEE Transactions on Intelligent Transportation Systems*, 20(5), 1828-1836.
- Gibran, K., & Bushrui, S. B. (2012). *The prophet: A new annotated edition*. Simon and Schuster.
- Godaheewa, R., Bergmeir, C., Webb, G. I., Hyndman, R. J., & Montero-Manso, P. (2021). Monash time series forecasting archive. *arXiv preprint arXiv:2105.06643*.
- Gong, Y., Liu, C., Zhang, C., Fu, L., Wei, H., & Pan, G. An Attention-Based Parallel Hybrid Prediction Model for Intersection Level Turning Movement Forecasting.
- Graves, A. (2012). Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, 37-45.
- Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural networks*, 18(5-6), 602-610.
- Gregor, K., Danihelka, I., Mnih, A., Blundell, C., & Wierstra, D. (2014). Deep autoregressive networks. International Conference on Machine Learning,
- Grigsby, J., Wang, Z., & Qi, Y. (2021). Long-range transformers for dynamic spatiotemporal forecasting. *arXiv preprint arXiv:2109.12218*.
- Gruver, N., Finzi, M., Qiu, S., & Wilson, A. G. (2024). Large language models are zero-shot time series forecasters. *Advances in neural information processing systems*, 36.
- Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN model-based approach in classification. On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings,
- Hamilton, J. D. (2020). *Time series analysis*. Princeton university press.
- Hamilton, W. L. (2020). *Graph representation learning*. Morgan & Claypool Publishers.
- Ho, T. K. (1995). Random decision forests. Proceedings of 3rd international conference on document analysis and recognition,
- Hochreiter, S., Bengio, Y., Frasconi, P., & Schmidhuber, J. (2001). Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In: A field guide to dynamical recurrent neural networks. IEEE Press In.
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735-1780.

- Hu, W., Yan, L., Liu, K., & Wang, H. (2016). A short-term traffic flow forecasting method based on the hybrid PSO-SVR. *Neural Processing Letters*, 43(1), 155-172.
- Hu, X. (2021). Stock price prediction based on temporal fusion transformer. 2021 3rd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI),
- Hu, X., Zhao, C., & Wang, G. (2020). A traffic light dynamic control algorithm with deep reinforcement learning based on GNN prediction. *arXiv preprint arXiv:2009.14627*.
- Huy, P. C., Minh, N. Q., Tien, N. D., & Anh, T. T. Q. (2022). Short-term electricity load forecasting based on temporal fusion transformer model. *IEEE Access*, 10, 106296-106304.
- Innamaa, S. (2000). Short-term prediction of traffic situation using MLP-neural networks. Proceedings of the 7th world congress on intelligent transport systems, Turin, Italy,
- Jiang, W., & Luo, J. (2022). Graph neural network for traffic forecasting: A survey. *Expert systems with applications*, 207, 117921.
- Johansen, T. A., & Foss, B. (1993). Constructing NARMAX models using ARMAX models. *International journal of control*, 58(5), 1125-1153.
- Kao, I.-F., Zhou, Y., Chang, L.-C., & Chang, F.-J. (2020). Exploring a Long Short-Term Memory based Encoder-Decoder framework for multi-step-ahead flood forecasting. *Journal of Hydrology*, 583, 124631.
- Karnati, Y., Sengupta, R., Rangarajan, A., & Ranka, S. (2021). Subcycle-based Neural Network Algorithms for Turning Movement Count Prediction. VEHITS,
- Ketkar, N. (2017). Introduction to keras. In *Deep learning with Python* (pp. 97-111). Springer.
- Khan, K., Rehman, S. U., Aziz, K., Fong, S., & Sarasvady, S. (2014). DBSCAN: Past, present and future. The fifth international conference on the applications of digital information and web technologies (ICADIWT 2014),
- Khan, M. A., Saqib, S., Alyas, T., Rehman, A. U., Saeed, Y., Zeb, A., Zareei, M., & Mohamed, E. M. (2020). Effective demand forecasting model using business intelligence empowered with machine learning. *IEEE Access*, 8, 116013-116023.
- Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Koenker, R. (2005). *Quantile regression* (Vol. 38). Cambridge university press.

- Kowsher, M., Tahabilder, A., Sanjid, M. Z. I., Prottasha, N. J., Uddin, M. S., Hossain, M. A., & Jilani, M. A. K. (2021). LSTM-ANN & BiLSTM-ANN: Hybrid deep learning models for enhanced classification accuracy. *Procedia Computer Science*, 193, 131-140.
- Kumar, K., Parida, M., & Katiyar, V. (2013). Short term traffic flow prediction for a non urban highway using artificial neural network. *Procedia-Social and Behavioral Sciences*, 104, 755-764.
- Kumar, S. V., & Vanajakshi, L. (2015). Short-term traffic flow prediction using seasonal ARIMA model with limited input data. *European Transport Research Review*, 7(3), 1-9.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, 521(7553), 436-444.
- Leshem, G., & Ritov, Y. a. (2007). Traffic flow prediction using adaboost algorithm with random forests as a weak learner. *International Journal of Mathematical and Computational Sciences*, 1(1), 1-6.
- Li, J., Guo, F., Sivakumar, A., Dong, Y., & Krishnan, R. (2021). Transferability improvement in short-term traffic prediction using stacked LSTM network. *Transportation research part C: emerging technologies*, 124, 102977.
- Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., & Yan, X. (2019). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems*, 32.
- Li, W., Ban, X. J., Zheng, J., Liu, H. X., Gong, C., & Li, Y. (2020). Real-time movement-based traffic volume prediction at signalized intersections. *Journal of Transportation Engineering, Part A: Systems*, 146(8), 04020081.
- Li, W., Wang, J., Fan, R., Zhang, Y., Guo, Q., Siddique, C., & Ban, X. J. (2020). Short-term traffic state prediction from latent structures: Accuracy vs. efficiency. *Transportation Research Part C: Emerging Technologies*, 111, 72-90.
- Li, Z., Xiong, G., Chen, Y., Lv, Y., Hu, B., Zhu, F., & Wang, F.-Y. (2019). A hybrid deep learning approach with GCN and LSTM for traffic flow prediction. 2019 IEEE intelligent transportation systems conference (ITSC),
- Liang, Y., & Qiu, L. (2015). Network traffic prediction based on SVR improved by chaos theory and ant colony optimization. *International journal of future generation communication and networking*, 8(1), 69-78.

- Liao, Y., & Liang, C. (2021). A temperature time series forecasting model based on DeepAR. 2021 7th International Conference on Computer and Communications (ICCC),
- Liaw, R., Liang, E., Nishihara, R., Moritz, P., Gonzalez, J. E., & Stoica, I. (2018). Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*.
- Lim, B., Arık, S. Ö., Loeff, N., & Pfister, T. (2021). Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4), 1748-1764.
- Lin, L., Handley, J. C., Gu, Y., Zhu, L., Wen, X., & Sadek, A. W. (2018). Quantifying uncertainty in short-term traffic prediction and its application to optimal staffing plan development. *Transportation Research Part C: Emerging Technologies*, 92, 323-348.
- Liu, Y., Qin, G., Huang, X., Wang, J., & Long, M. (2024). Autotimes: Autoregressive time series forecasters via large language models. *arXiv preprint arXiv:2402.02370*.
- Liu, Y., & Wu, H. (2017). Prediction of road traffic congestion based on random forest. 2017 10th International Symposium on Computational Intelligence and Design (ISCID),
- Luo, X., Li, D., & Zhang, S. (2019). Traffic flow prediction during the holidays based on DFT and SVR. *Journal of Sensors*, 2019.
- Ma, X., Dai, Z., He, Z., Ma, J., Wang, Y., & Wang, Y. (2017). Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. *Sensors*, 17(4), 818.
- Ma, Y., Zhang, Z., & Ihler, A. (2020). Multi-lane short-term traffic forecasting with convolutional LSTM network. *IEEE Access*, 8, 34629-34643.
- Mahdavian, A., Shojaei, A., Salem, M., Laman, H., Eluru, N., & Oloufa, A. A. (2021). A Universal Automated Data-Driven Modeling Framework for Truck Traffic Volume Prediction. *IEEE Access*, 9, 105341-105356.
- Mahmoud, N., Abdel-Aty, M., Cai, Q., & Yuan, J. (2022). Estimating cycle-level real-time traffic movements at signalized intersections. *Journal of Intelligent Transportation Systems*, 26(4), 400-419.
- Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2020). The M4 Competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1), 54-74.

- McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4), 115-133.
- Müller, M. (2007). Dynamic time warping. *Information retrieval for music and motion*, 69-84.
- Nazir, A., Shaikh, A. K., Shah, A. S., & Khalil, A. (2023). Forecasting energy consumption demand of customers in smart grid using Temporal Fusion Transformer (TFT). *Results in Engineering*, 17, 100888.
- Nie, Y., Nguyen, N. H., Sinthong, P., & Kalagnanam, J. (2022). A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*.
- Oreshkin, B. N., Carпов, D., Chapados, N., & Bengio, Y. (2019). N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*.
- Pan, C., Zhu, J., Kong, Z., Shi, H., & Yang, W. (2021). DC-STGCN: Dual-channel based graph convolutional networks for network traffic forecasting. *Electronics*, 10(9), 1014.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., & Antiga, L. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.
- Plan, A. S. (2016). 2016-2020. In: URL.
- Popescu, M.-C., Balas, V. E., Perescu-Popescu, L., & Mastorakis, N. (2009). Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, 8(7), 579-588.
- Pourpanah, F., Abdar, M., Luo, Y., Zhou, X., Wang, R., Lim, C. P., Wang, X.-Z., & Wu, Q. J. (2022). A review of generalized zero-shot learning methods. *IEEE transactions on pattern analysis and machine intelligence*, 45(4), 4051-4070.
- Qin, Y., Luo, H., Zhao, F., Fang, Y., Tao, X., & Wang, C. (2023). Spatio-temporal hierarchical MLP network for traffic forecasting. *Information Sciences*, 632, 543-554.
- Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, 1(1), 81-106.
- Rahman, R., Zhang, J., Tirtha, S. D., Bhowmik, T., Jahan, I., Eluru, N., & Hasan, S. (2022). A data-driven network model for traffic volume prediction at signalized intersections. *Journal of big data analytics in transportation*, 1-18.
- Ran, X., Shan, Z., Fang, Y., & Lin, C. (2019). An LSTM-based method with attention mechanism for travel time prediction. *Sensors*, 19(4), 861.

- Rasul, K., Ashok, A., Williams, A. R., Khorasani, A., Adamopoulos, G., Bhagwatkar, R., Biloš, M., Ghonia, H., Hassen, N., & Schneider, A. (2023). Lag-llama: Towards foundation models for time series forecasting. R0-FoMo: Robustness of Few-shot and Zero-shot Learning in Large Foundation Models,
- Rigatti, S. J. (2017). Random forest. *Journal of Insurance Medicine*, 47(1), 31-39.
- Rilling, G., Flandrin, P., & Goncalves, P. (2003). On empirical mode decomposition and its algorithms. IEEE-EURASIP workshop on nonlinear signal and image processing,
- Salinas, D., Flunkert, V., Gasthaus, J., & Januschowski, T. (2020). DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3), 1181-1191.
- Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2008). The graph neural network model. *IEEE transactions on neural networks*, 20(1), 61-80.
- Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11), 2673-2681.
- Segal, M. R. (2004). Machine learning benchmarks and random forest regression.
- Sharma, R., Agarwal, V., Sharma, S., & Arya, M. S. (2021). An LSTM-Based Fake News Detection System Using Word Embeddings-Based Feature Extraction. In *ICT Analysis and Applications* (pp. 247-255). Springer.
- Sherstinsky, A. (2020). Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404, 132306.
- Smith, B. L., & Demetsky, M. J. (1997). Traffic flow forecasting: comparison of modeling approaches. *Journal of transportation engineering*, 123(4), 261-266.
- Smith, B. L., Williams, B. M., & Oswald, R. K. (2002). Comparison of parametric and nonparametric models for traffic flow forecasting. *Transportation Research Part C: Emerging Technologies*, 10(4), 303-321.
- Spears, J., & Kalinowski, T. (2010). Toronto commuting times worst of 19 major cities. *The Toronto Star*. Retrieved March, 25, 2010.
- Sun, B., Cheng, W., Goswami, P., & Bai, G. (2018). Short-term traffic forecasting using self-adjusting k-nearest neighbours. *IET Intelligent Transport Systems*, 12(1), 41-48.
- Sutskever, I. (2014). Sequence to Sequence Learning with Neural Networks. *arXiv preprint arXiv:1409.3215*.

- Taylor, S. J., & Letham, B. (2018). Forecasting at scale. *The American Statistician*, 72(1), 37-45.
- Tian, Y., & Pan, L. (2015). Predicting short-term traffic flow by long short-term memory recurrent neural network. 2015 IEEE international conference on smart city/SocialCom/SustainCom (SmartCity),
- Tian, Y., Zhang, K., Li, J., Lin, X., & Yang, B. (2018). LSTM-based traffic flow prediction with missing data. *Neurocomputing*, 318, 297-305.
- Tong, M., & Xue, H. (2008). Highway traffic volume forecasting based on seasonal ARIMA model. *Journal of Highway and Transportation Research and Development (English Edition)*, 3(2), 109-112.
- Van Lint, J., Hoogendoorn, S., & van Zuylen, H. J. (2002). Freeway travel time prediction with state-space neural networks: modeling state-space dynamics with recurrent neural networks. *Transportation Research Record*, 1811(1), 30-39.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.
- Wang, C., Quddus, M. A., & Ison, S. G. (2013). The effect of traffic and road characteristics on road safety: A review and future research direction. *Safety science*, 57, 264-275.
- Wang, D., Meng, Y., Chen, S., Xie, C., & Liu, Z. (2021). A Hybrid Model for Vessel Traffic Flow Prediction Based on Wavelet and Prophet. *Journal of Marine Science and Engineering*, 9(11), 1231.
- Wang, Y., Ban, X. J., Cui, Z., & Zhu, M. (2019). An artificial intelligence platform for network-wide congestion detection and prediction using multi-source data.
- Wang, Z., Su, X., & Ding, Z. (2020). Long-term traffic prediction based on lstm encoder-decoder architecture. *IEEE Transactions on Intelligent Transportation Systems*, 22(10), 6561-6571.
- Waterloo, R. o. (2022). Region of Waterloo Open Data.
<https://www.regionofwaterloo.ca/en/regional-government/open-data.aspx>.
- Wei, D., & Liu, H. (2013). An adaptive-margin support vector regression for short-term traffic flow forecast. *Journal of Intelligent Transportation Systems*, 17(4), 317-327.

- Williams, B. M., & Hoel, L. A. (2003). Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *Journal of transportation engineering*, 129(6), 664-672.
- Wu, J., & Thnay, C. (2001). An O–D based method for estimating link and turning volume based on counts. *ITE District*, 6, 8-11.
- Wu, Q., He, K., Chen, X., Yu, S., & Zhang, J. (2021). Deep transfer learning across cities for mobile traffic prediction. *IEEE/ACM Transactions on Networking*, 30(3), 1255-1267.
- Xie, Y., Xiong, Y., & Zhu, Y. (2020). SAST-GNN: a self-attention based spatio-temporal graph neural network for traffic prediction. Database Systems for Advanced Applications: 25th International Conference, DASFAA 2020, Jeju, South Korea, September 24–27, 2020, Proceedings, Part I 25,
- Xu, M., Dai, W., Liu, C., Gao, X., Lin, W., Qi, G.-J., & Xiong, H. (2020). Spatial-temporal transformer networks for traffic flow forecasting. *arXiv preprint arXiv:2001.02908*.
- Yan, H., Ma, X., & Pu, Z. (2021). Learning dynamic and hierarchical traffic spatiotemporal features with transformer. *IEEE Transactions on Intelligent Transportation Systems*, 23(11), 22386-22399.
- Yan, X., & Su, X. (2009). *Linear regression analysis: theory and computing*. world scientific.
- Yan, Z. (2010). Traj-ARIMA: A spatial-time series model for network-constrained trajectory. Proceedings of the Third International Workshop on Computational Transportation Science,
- Yenduri, G., Ramalingam, M., Selvi, G. C., Supriya, Y., Srivastava, G., Maddikunta, P. K. R., Raj, G. D., Jhaveri, R. H., Prabadevi, B., & Wang, W. (2024). Gpt (generative pre-trained transformer)—a comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions. *IEEE Access*.
- Yu, B., Lee, Y., & Sohn, K. (2020). Forecasting road traffic speeds by considering area-wide spatio-temporal dependencies based on a graph convolutional neural network (GCN). *Transportation research part C: emerging technologies*, 114, 189-204.
- Yu, B., Yin, H., & Zhu, Z. (2017). Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*.
- Zar, J. H. (2005). Spearman rank correlation. *Encyclopedia of biostatistics*, 7.

- Zhang, C., Pan, G., & Fu, L. (2023). Real-Time Intersection Turning Movement Flow Forecasting Using a Parallel Bidirectional Long Short-Term Memory Neural Network Model. *Transportation Research Record*, 03611981231172958.
- Zhang, C., Pan, G., & Fu, L. (2024). Real-Time Intersection Turning Movement Flow Forecasting Using a Parallel Bidirectional Long Short-Term Memory Neural Network Model. *Transportation Research Record*, 2678(2), 167-183.
<https://doi.org/10.1177/03611981231172958>
- Zhang, C., Pan, G., Muresan, M. I., & Fu, L. (2024). *A DEEP LEARNING MODEL FOR INTERSECTION-LEVEL TURNING MOVEMENT FLOWS FORECASTING WITH UNCERTAINTY QUANTIFICATION* The 2024 Transportation Research Board (TRB) Annual Meeting, Washington, D.C.
- Zhang, C., Pan, G., Muresan, M. I., Lu, Z., & Fu, L. (2024). Temporal Fusion Transformer for Real-Time Intersection Turning Movement Flow Forecasting Incorporating Exogenous Factors. 2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC),
- Zhang, C., You, Y., Pan, G., Muresan, M. I., & Fu, L. (2025). Real-Time Traffic Turning Flow Forecasting Under Few-Shot and Zero-Shot Scenarios by Decoder-only Model. In U. o. Waterloo (Ed.).
- Zhang, G. P. (2003). Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*, 50, 159-175.
- Zhang, L., Liu, Q., Yang, W., Wei, N., & Dong, D. (2013). An improved k-nearest neighbor model for short-term traffic flow prediction. *Procedia-Social and Behavioral Sciences*, 96, 653-662.
- Zhang, W., Yu, Y., Qi, Y., Shu, F., & Wang, Y. (2019). Short-term traffic flow prediction based on spatio-temporal analysis and CNN deep learning. *Transportmetrica A: Transport Science*, 15(2), 1688-1711.
- Zhang, Z., Yang, H., & Yang, X. (2023). A Transfer Learning–Based LSTM for Traffic Flow Prediction with Missing Data. *Journal of Transportation Engineering, Part A: Systems*, 149(10), 04023095.
- Zhao, F., Dong, B., Sun, Y., & Huang, S. (2023). Analysis and Forecast of Railway Freight Volume based on Prophet-Deep AR Model. *Tehnički vjesnik*, 30(4), 1126-1134.

- Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., Deng, M., & Li, H. (2019). T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, 21(9), 3848-3858.
- Zhao, Z., Chen, W., Wu, X., Chen, P. C., & Liu, J. (2017). LSTM network: a deep learning approach for short-term traffic forecast. *IET Intelligent Transport Systems*, 11(2), 68-75.
- Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., & Zhang, W. (2021). Informer: Beyond efficient transformer for long sequence time-series forecasting. Proceedings of the AAAI conference on artificial intelligence,

Appendix

A.1 Python code for PB-LSTM Model

```
# Required Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import holidays
import tensorflow as tf
from keras import Input, Model
from keras.models import load_model
from keras.layers import LSTM, Dropout, Bidirectional, Dense, TimeDistributed, Flatten, concatenate,
BatchNormalization
from sklearn.preprocessing import MinMaxScaler

# Mount Drive
from google.colab import drive
drive.mount('/content/drive')

# Load dataset
dataset = pd.read_csv(dataset.csv')

# Create time series dataset function
def create_dataset(dataset, look_back, look_forward):
    dataX, dataY = [], []
    for i in range(len(dataset) - look_back - look_forward):
        dataX.append(dataset[i:(i + look_back)])
        dataY.append(dataset[i + look_back + (look_forward - 1)])
    return np.array(dataX).astype('float32'), np.array(dataY).astype('float32')

# Prepare training
index = list(set(dataset.ID_x))
look_forward = 1
look_back = 6
for i in range(0, len(index), 3):
    train_df = dataset[dataset['ID_x'] == index[i]]
    Time_index = train_df.Time_in[look_back:-1]
    train_df = train_df.drop(columns=['Time_in'])
    train_df = train_df.to_numpy()
    trainX, trainY = create_dataset(train_df, look_back, look_forward)

    trainX_time = trainX[:, :, 0:12]
    trainX_stamp = trainY[:, 12:]
    trainY_2 = trainY[:, 0:12]
    if i == 0:
```

```

# Define model
Timestamp = Input(shape=(17,), name='Timestamp')
Time = Input(shape=(6, 12), name='Time')
with tf.device('/device:GPU:0'):
    x1 = Dense(8, activation="relu")(Timestamp)
    x1 = BatchNormalization()(x1)
    x1 = Dense(16, activation="relu")(x1)
    x1 = BatchNormalization()(x1)
    x1 = Dense(32, activation="relu")(x1)
    x1 = BatchNormalization()(x1)
    x1 = Flatten()(x1)
with tf.device('/device:GPU:0'):
    x2 = TimeDistributed(Dense(12, activation="relu"))(Time)
    x2 = BatchNormalization()(x2)
    x2 = Bidirectional(LSTM(16, return_sequences=True))(x2)
    x2 = BatchNormalization()(x2)
    x2 = Bidirectional(LSTM(16, return_sequences=True))(x2)
    x2 = BatchNormalization()(x2)
    x2 = Bidirectional(LSTM(16, return_sequences=False))(x2)
    x2 = BatchNormalization()(x2)
    x2 = Flatten()(x2)
x = concatenate([x1, x2])
output_ = Dense(12, activation='relu')(x)
model = Model(inputs=[Timestamp, Time], outputs=[output_])
model.summary()

# Save model checkpoint
checkpoint_filepath = 'MIM_model.h5'
model_checkpoint_callback = tf.keras.callbacks.ModelCheckpoint(
    filepath=checkpoint_filepath,
    save_weights_only=False,
    monitor='val_loss',
    mode='min',
    save_best_only=True,
    verbose=1
)

# Compile & train
model.compile(loss='mean_absolute_percentage_error', optimizer='adam')
model.fit(
    [trainX_stamp, trainX_time], trainY_2,
    epochs=100, batch_size=2016, verbose=1,
    validation_split=0.2, shuffle=True,
    callbacks=[model_checkpoint_callback]
)
else:

```

```

# Fine-tune for other IDs
model.fit(
    [trainX_stamp, trainX_time], trainY_2,
    epochs=100, batch_size=2016, verbose=1,
    validation_split=0.2, shuffle=True,
    callbacks=[model_checkpoint_callback]
)

# Forecast
trainPredict = model.predict([trainX_stamp, trainX_time])

```

A.2 Python code for DeepAR Model

```

# Required Libraries
import numpy as np
import pandas as pd
import torch
import matplotlib.pyplot as plt
import holidays
import lightning.pytorch as pl
from lightning.pytorch.callbacks import ModelCheckpoint, LearningRateMonitor
from pytorch_forecasting import (
    TimeSeriesDataSet, DeepAR, NaNLabelEncoder, Baseline,
    MultivariateNormalDistributionLoss, SMAPE
)
from pytorch_lightning.tuner import Tuner

# Mount Drive
from google.colab import drive
drive.mount('/content/drive')

# Load dataset
df = pd.read_csv('dataset.csv')

# Define sequence lengths
max_encoder_length = 96
max_prediction_length = 1
training_cutoff = df['time_idx'].max() - max_prediction_length

# Create time series dataset
training = TimeSeriesDataSet(
    df[df.time_idx <= training_cutoff],
    time_idx="time_idx",
    target="Sumin",
    group_ids=["group"],
    max_encoder_length=max_encoder_length,
    max_prediction_length=max_prediction_length,
    time_varying_unknown_reals=["Sumin"],

```

```

static_categoricals=["weekday", "holiday", "hour", "year", "month"],
categorical_encoders={
    col: NaNLabelEncoder().fit(df[col]) for col in ["weekday", "holiday", "hour", "year", "month"]
},
allow_missing_timesteps=True)

# Validation dataset
validation = TimeSeriesDataSet.from_dataset(training, df, min_prediction_idx=training_cutoff + 1)

# Dataloaders
train_loader = training.to_dataloader(train=True, batch_size=128, batch_sampler="synchronized")
val_loader = validation.to_dataloader(train=False, batch_size=128, batch_sampler="synchronized")

# Evaluate baseline performance
baseline_preds = Baseline().predict(val_loader, return_y=True, trainer_kwargs={"accelerator": "gpu"})
print("Baseline SMAPE:", SMAPE()(baseline_preds.output, baseline_preds.y))

# Trainer and tuner setup
pl.seed_everything(42)
trainer = pl.Trainer(accelerator="gpu", gradient_clip_val=0.1)
net = DeepAR.from_dataset(
    training,
    learning_rate=3e-2,
    hidden_size=30,
    rnn_layers=2,
    loss=MultivariateNormalDistributionLoss(rank=30),
    optimizer="Adam"
)

# Learning rate finder
tuner = Tuner(trainer)
lr_finder = tuner.lr_find(net, train_dataloaders=train_loader, val_dataloaders=val_loader)
suggested_lr = lr_finder.suggestion()
net.hparams.learning_rate = suggested_lr
print(f'Suggested learning rate: {suggested_lr:.2E}')

# Save model checkpoint
checkpoint_callback = ModelCheckpoint(
    monitor='val_loss',
    filename='deepar_best_epoch{epoch:02d}-val{val_loss:.2f}',
    save_last=True,
    save_weights_only=True,
    auto_insert_metric_name=True,
    mode='min'
)
lr_logger = LearningRateMonitor()

# Final trainer config
trainer = pl.Trainer(
    max_epochs=1000,

```

```

    min_epochs=800,
    accelerator="gpu",
    devices="auto",
    limit_train_batches=30,
    limit_val_batches=3,
    enable_model_summary=True,
    callbacks=[checkpoint_callback, lr_logger]
)

# Rebuild model with updated learning rate
deepar = DeepAR.from_dataset(
    training,
    learning_rate=suggested_lr,
    hidden_size=30,
    rnn_layers=2,
    loss=MultivariateNormalDistributionLoss(rank=30),
    optimizer="Adam"
)
print(f'Number of parameters: {deepar.size()/1e3:.1f}k')
trainer.fit(deepar, train_dataloaders=train_loader, val_dataloaders=val_loader)

# Load best checkpoint and evaluate
best_model_path = checkpoint_callback.best_model_path
best_model = DeepAR.load_from_checkpoint(best_model_path)

# Forecast
raw_preds = best_model.predict(val_loader, mode="raw", return_x=True, trainer_kwargs={"accelerator":
"gpu"})

```

A.3 Python code for TFT Model

```

# Required Libraries
import pandas as pd
import torch
import time
import holidays
import pickle
import warnings
import lightning.pytorch as pl
import matplotlib.pyplot as plt
from pytorch_forecasting import (
    TemporalFusionTransformer, TimeSeriesDataSet, GroupNormalizer,
    NaNLabelEncoder, QuantileLoss
)
from lightning.pytorch.callbacks import (
    ModelCheckpoint, LearningRateMonitor, EarlyStopping
)

# Mount Drive

```

```

from google.colab import drive
drive.mount('/content/drive')

# Load dataset
df = pd.read_csv('dataset.csv')

# Define sequence lengths
max_encoder_length = 288
max_prediction_length = 288
training_cutoff = df['time_idx'].max() - max_prediction_length

# Create time series dataset
training = TimeSeriesDataSet(
    df[lambdax: x.time_idx <= training_cutoff],
    time_idx='time_idx',
    target='Count',
    group_ids=['ID_x', 'Movement'],
    static_reals=['SpeedZone', 'NumberofLanes'],
    static_categoricals=['CartoClass', 'FlowDirection', 'DividedRoad', 'DBSCAN', 'Labels'],
    time_varying_known_reals=['VISIBILITY', 'PRECIP_AMOUNT', 'TEMP'],
    time_varying_known_categoricals=['WEATHER_ENG_DESC', 'weekday', 'hour', 'year', 'month',
'holiday'],
    time_varying_unknown_reals=['Count'],
    min_encoder_length=max_encoder_length // 2,
    max_encoder_length=max_encoder_length,
    min_prediction_length=1,
    max_prediction_length=max_prediction_length,
    target_normalizer=GroupNormalizer(groups=['ID_x', 'Movement'], transformation='softplus'),
    allow_missing_timesteps=True,
    add_relative_time_idx=True,
    add_target_scales=True,
    add_encoder_length=True
)

# Validation dataset
validation = TimeSeriesDataSet.from_dataset(training, df, predict=True, stop_randomization=True)

# Dataloaders
train_loader = training.to_dataloader(train=True, batch_size=256, batch_sampler="synchronized",
num_workers=4)
val_loader = validation.to_dataloader(train=False, batch_size=256 * 10, batch_sampler="synchronized",
num_workers=4)

# Trainer and tuner setup
checkpoint_callback = ModelCheckpoint(
    monitor='val_loss',
    dirpath='.', # <-- save to current directory or specify path
    filename="TFT_model",
    save_last=True,
    save_weights_only=True,

```

```

        auto_insert_metric_name=True,
        mode='min'
    )
    lr_logger = LearningRateMonitor()
    early_stop_callback = EarlyStopping(monitor="val_loss", patience=5, verbose=True, mode="min")
    trainer = pl.Trainer(
        max_epochs=200,
        min_epochs=100,
        accelerator="gpu",
        devices="auto",
        gradient_clip_val=0.1,
        limit_train_batches=30,
        limit_val_batches=3,
        enable_model_summary=True,
        callbacks=[lr_logger, checkpoint_callback, early_stop_callback]
    )

    #Model definition and training
    start_time = time.time()
    tft = TemporalFusionTransformer.from_dataset(
        training,
        learning_rate=1e-2,
        hidden_size=60,
        lstm_layers=2,
        attention_head_size=4,
        dropout=0.1,
        loss=QuantileLoss(),
        log_interval=10,
        log_val_interval=1,
        weight_decay=1e-2,
        reduce_on_plateau_patience=10,
        output_size=7 # quantiles
    )
    print(f"Model size: {tft.size() / 1e3:.1f}k parameters")
    trainer.fit(tft, train_dataloaders=train_loader, val_dataloaders=val_loader)
    print("Training time (s):", time.time() - start_time)

    # Load best checkpoint and evaluate
    best_model_path = checkpoint_callback.best_model_path
    print("Best model path:", best_model_path)
    best_tft = TemporalFusionTransformer.load_from_checkpoint(best_model_path)

    #Forecasting
    raw_preds = best_tft.predict(
        training.filter(lambda x: (x.Movement == "EBL") & (x.time_idx_first_prediction == 100000)),
        mode="raw", return_x=True
    )

    for idx in range(8):
        best_tft.plot_prediction(raw_preds.x, raw_preds.output, idx=idx, add_loss_to_title=True)

```

```

# Variable-level analysis
predictions = best_tft.predict(
    training.filter(lambda x: (x.Movement == "EBL") & (x.time_idx_first_prediction == 100000)),
    return_x=True
)
var_analysis = best_tft.calculate_prediction_actual_by_variable(predictions.x, predictions.output)
best_tft.plot_prediction_actual_by_variable(var_analysis)

# Attention interpretation
interpretation = best_tft.interpret_output(raw_preds.output, reduction="sum")
best_tft.plot_interpretation(interpretation)

```

A.4 Python code for TimesFM Model

```

# Required Libraries
import timesfm
import pandas as pd
import numpy as np
import time
from matplotlib import pyplot as plt

# Initialize TimesFM
tfm = timesfm.TimesFm(
    hparams=timesfm.TimesFmHparams(
        backend="gpu",
        per_core_batch_size=32,
        horizon_len=96,
        context_len=96
    ),
    checkpoint=timesfm.TimesFmCheckpoint(
        huggingface_repo_id="google/timesfm-1.0-200m-pytorch"
    )
)

# Load datasets
ds_og = pd.read_csv("Train_TFT_MI_15mins_12.csv")
ds_decomW = pd.read_excel("MI_12_decom.xlsx")
ds_decomD = pd.read_excel("MI_12_decom_W.xlsx")

# Create time series dataset
def data_preprocess(movement: str, ds_og: pd.DataFrame, ds_decom: pd.DataFrame) -> pd.DataFrame:
    ds_og = ds_og[ds_og['Movement'] == movement]
    ds_decom = ds_decom.loc[:, ds_decom.columns.str.contains(movement)].copy()
    ds_decom['time_idx'] = ds_og['time_idx'].values
    ds_decom['Time_in'] = pd.to_datetime(ds_og['Time_in'].values, utc=True)
    ds_decom['Time_in'] = ds_decom['Time_in'].dt.tz_convert('Canada/Eastern')
    melted_df = pd.melt(
        ds_decom,

```

```

        id_vars=['Time_in', 'time_idx'],
        value_vars=ds_decom.columns[:-2],
        var_name='Movement',
        value_name='Count'
    )
    return melted_df

# Export observed series for each movement
movements = ['EBL', 'EBR', 'EBT', 'NBL', 'NBR', 'NBT',
             'SBL', 'SBR', 'SBT', 'WBL', 'WBR', 'WBT']
for mv in movements:
    local_ds = data_preprocess(mv, ds_og, ds_decomW)
    local_ds.to_csv(f'result/TimesFM_{mv}_observed.csv', index=False)

# Forecast using daily decomposed input (ds_decomD)
movement = 'EBT'
local_ds = data_preprocess(movement, ds_og, ds_decomD)
local_ds = local_ds[local_ds['Movement'] != f'{movement}_original']
start_time = time.time()
forecast = tfm.forecast_on_df(
    inputs=local_ds.rename(columns={'Movement': 'unique_id', 'Time_in': 'ds'}),
    freq="15min",
    value_name='Count',
    num_jobs=1
).rename(columns={'unique_id': 'Movement', 'ds': 'Time_in'})
end_time = time.time()
forecast.to_csv(f'result/TimesFM_{movement}_decomD.csv', index=False)
print(f'{movement} (daily decomposed) forecast completed in {end_time - start_time:.2f} seconds")

```