The Prediction and Mitigation of Road Traffic Congestion Based on Machine Learning

Z E BARTLETT

PhD 2022

DOCTORAL THESIS

The Prediction and Mitigation of Road Traffic Congestion Based on Machine Learning

Author:
Zoe Bartlett-Giagos

A thesis submitted in fulfilment of the requirements of

MANCHESTER METROPOLITAN UNIVERSITY

for the degree of Doctor of Philosophy

in the

Centre for Advanced Computational Science
Department of Computing and Mathematics

2022

Declaration of Authorship

I, Zoe Bartlett-Giagos, declare that this thesis titled, "The Prediction and Mitigation of Road Traffic Congestion Based on Machine Learning" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

Abstract

Traffic congestion is a major issue for all developed countries. In most urbanised areas space is a scarce commodity. Therefore, better management of the existing roads to increase or maintain their capacity level is the only viable solution. Research in the last two decades has focused on Intelligent Transport Systems (ITS) development. Predicting traffic flow in real time can be used to prevent or alleviate future congestion. The key to an effective proactive method is a model that produces timely and accurate predictions. However, despite extensive research in this area, a reliable method is still not available. Therefore, in this thesis, we developed an accurate online road traffic flow prediction model, with a particular focus on heterogeneous traffic flow, for urbanised road networks. The contributions of this work include:

Firstly, we conducted a comprehensive literature review and benchmark evaluation of existing machine learning models using a real dataset obtained from Transport for Greater Manchester. We investigated their prediction accuracy, time horizon sensitivity, and input feature settings (different classes of vehicles), to understand how they can affect their prediction accuracy. The experimental results show that the artificial neural network was the most successful at predicting short-term road traffic flow. Additionally, it was found that different classes of vehicles can improve prediction accuracy.

Secondly, we examined three recurrent neural networks (a standard recurrent, a long short-term memory, and a gated recurrent unit). We compared their accuracy, training time, and sensitivity to architectural change using a new performance metric we developed to standardise the accuracy and training time into a comparable score (STATS). The experimental results show that the gated recurrent unit performed the best and was most stable against architectural changes. Conversely, the long short-term memory was the least stable model.

Thirdly, we investigated different magnitudes of temporal patterns in the dataset, both short and long-term, to understand how contextual temporal data can improve prediction accuracy. We also developed a novel online dynamic temporal context neural network framework. The framework dynamically determines how useful a temporal data segment is for prediction, and weights it accordingly for use in the regression model. The experimental results show that short and long-term temporal patterns improved prediction accuracy. In addition, the proposed online dynamical framework improved prediction results by 10.8% when compared with a deep gated recurrent model.

Finally, we investigated the dynamic nature of road traffic flow's input features by examining their spatial and temporal relationships. We also developed a novel dynamic exogenous feature filter mechanism. The feature filter mechanism uses 'local windows' to filter input features in real-time to improve prediction accuracy. The results show that a global correlation was insufficient to describe the complex and dynamic relationships between the input features. The local correlations (local windows) were able to identify additional geospatial and temporal relationships. Furthermore, the proposed feature filter mechanism was compared to a state-of-the-art method, a dynamic rolling window feature filter model. The experimental results showed that the proposed model was the most accurate, with an RMSE of 10.06%, closely followed by the dynamic rolling window feature filter model, with an RMSE of 10.98%. However, the proposed model was computationally much lighter than the rolling windows model.

Acknowledgements

First, I would like to thank my Director of Studies, Prof. Liangxiu Han, for their invaluable support and patience over the years. I am deeply grateful for their guidance. Secondly, I would also like to express my sincere gratitude to Prof. Trung Thanh Nguyen and Dr Princy Johnson for their insightful comments, which have greatly influenced my work. Thirdly, I would like to thank my colleagues that became lifelong friends. Dr Lewis Evans, Dr Kristopher Welsh, and Dr Amy Khalfrey; an infinite sources of wisdom and good humour. Lastly, I thank my loving husband, Dr Vasileios Giagos, for his understanding, encouragement, and late-night proofreading.

Author's Publications

Z. Bartlett, L. Han, T.T. Nguyen, and P. Johnson. A Novel Online Dynamic Temporal Context Neural Network framework for the Prediction of Road Traffic Flow. IEEE Access, 7, 2019. ISSN 21693536. doi: 10.1109/ACCESS.2019.2943028.

Z. Bartlett, L. Han, T.T. Nguyen, and P. Johnson. Prediction of Road Traffic Flow Based on Deep Recurrent Neural Networks. In The 5th IEEE Smart World Congress, Leicester, UK, 2019. IEEE.

Z. Bartlett, L. Han, T.T. Nguyen, and P. Johnson. A Machine Learning Based Approach for the Prediction of Road Traffic Congestion. In 16th International Conference on Smart City 2018, Exeter, UK, 2019. IEEE.

Z. Bartlett, L. Han, T.T. Nguyen, and P. Johnson. A Novel Dynamic Exogenous Feature Filter using Local Windows for Deep Learning-based Prediction of Road Traffic Flow. Submitted to IEEE ACCESS (Under Review).

Z. Bartlett, L. Han, T.T. Nguyen, and P. Johnson. A Dynamic Bayesian Model to Assess the Affect of a Large Scale Non-Recurrent Road Traffic Incidents. In preparation.

Contents

List of Figures

List of Tables

List of Abbreviations

AE	**A**bsolute **E**rror
AE	**A**uto**e**ncoder
AID	**A**utomatic **I**ncident **D**etection
ANN	**A**rtificial **N**eural **N**etwork
APSO	**A**daptive **P**article **S**warm **O**ptimisation
ARCH	**A**uto**r**egressive **C**onditional **H**eteroskedasticity
ARIMA	**A**uto**r**egressive **I**ntegrated **M**oving **A**verages
APE	**A**verage **P**ercentage **E**rror
BGD	**B**atch **G**radient **D**escent
BPNN	**B**ack**p**ropergation **N**eural **N**etwork
CAP	**C**redit **A**ssignment **P**ath
CCF	**C**ross **C**orrelation **F**unction
CEC	**C**onstant **E**rror **C**arousel
CNN	**Convolutional** **N**eural **N**etwork
ConvLSTM	**Conv**olutional **L**ong **S**hort **T**erm **M**emory
CPSD	**C**onditional **P**redicted **S**tandard **D**eviation
DeepST	**Deep** Learning Based Prediction Model for **S**patial and **T**emporal Data
DL	**D**eep **L**earning
DNN	**D**eep **N**eural **N**etworks
DR	**D**etection **R**ate
DTA	**D**ynamic **T**raffic **A**ssignment
DTC	**D**ynamic **T**emporal **C**ontext
DYNASMART-X	**Dyna**mic Traffic Assignment **S**imulation **M**odel for **A**davanced **R**oad **T**elematics
FAR	**F**alse **A**larm **R**ate
FFNN	**F**eed **F**orward **N**eural **N**etwork
FS	**F**eature **S**election
FSM	**F**eature **S**election **M**ethod

GAENN	**G**raph **A**uto**e**ncoder **N**eural **N**etwork
GANN	**G**raph **A**ttention **N**eural **N**etwork
GARCH	**G**eneralised **A**uto**r**egressive **C**onditional **H**eteroskedasticity
GCGRU	**G**raph **C**onvolution **G**ated **R**ecurrent **U**nit
GCNN	**G**raph **C**onvolutional **N**eural **N**etwork
GD	**G**radient **D**escent
GFS	**G**rey **F**orecasting **S**ystem
GGNN	**G**raph **G**enerative **e**ncoder **N**eural **N**etwork
GM	**G**rey **M**odel
GNN	**G**raph **N**eural **N**etwork
GRNN	**G**raph **R**ecurrent **N**eural **N**etwork
GSM	**G**rey **S**ystem **M**odels
GRUNN	**G**ated **R**ecurrent **U**nit **N**eural **N**etwork
HGV	**H**eavy **G**oods **V**ehicle
IDW	**I**nverse **D**istance **W**eighting
ILD	**I**nductive **L**oop **D**etector
ITS	**I**nteligent **T**ransport **S**ystem
JSNN	**J**ordan **S**equential **N**eural **N**etwork
KF	**K**alman **F**ilter
KNN	**K** **N**earest **N**eighbours
LASSO	**L**east **A**bsolute **S**hrinkage and **S**election **O**perator
LCAP	**L**ondon **C**ongestion **A**nalysis **P**roject
LSTM	**L**ong **S**hort **T**erm **M**emory
LSTMNN	**L**ong **S**hort **T**erm **M**emory **N**eural **N**etwork
LTA	**L**and **T**ransport **A**uthority
MAE	**Mean** **A**verage **E**rror
MAPE	**Mean** **A**verage **P**ercentage **E**rror
ML	**M**achine **L**earning
MAD	**M**ean **A**bsolute **D**eviation
MARE	**M**ean **A**bsolute **R**elative **E**rror
MRE	**M**ean **R**elative **E**rror
MSE	**M**ean **S**quared **E**rror
NARX	**N**on-linear **A**uto**r**egressive Neural Network with E**x**ogenous Inputs
OGCRNN	**O**ptimised **G**raph **C**onvolutional **R**ecurrent **N**eural **N**etworks

ODC	**O**rigin **D**estination **C**ost
OF	**O**ptical **F**low
OGM	**O**ptimised **G**raph **M**atrices
PBC	**P**earson's **B**ivariate **C**orrelation
PC	**P**rinciple **C**omponet
PCA	**P**rinciple **C**omponet **A**nalysis
PeMS	**Pe**rformance **M**easurement **S**ystems
PSD	**P**redicted **S**tandard **D**eviation
PSO	**P**article **S**warm **O**ptimisation
RBF	**R**adial **B**asis **F**unction
RE	**R**elative **E**rror
ReLU	**Re**ctified **L**inear **U**nit
RMSE	**R**oot **M**ean **S**quared **E**rror
RNN	**R**ecurrent **N**eural **N**etwork
RPCA	**R**obust **P**rinciple **C**omponet **A**nalysis
RTI	**R**oad **T**raffic **I**ncident
RTSS	**R**oad **T**rafiic **S**ensor **S**ite
SAE	**Stacked A**uto**e**ncoder
SARIMA	**S**easonal **A**uto **R**egressive **I**ntegrated **M**oving **A**verages
SGD	**S**tochastic **G**radient **D**escent
SMART	**S**calable **M**icroscopic **A**daptive **R**oad **T**raffic Simulator
SORT	**S**imple **O**nline **R**eal-time **T**racking
SRCC	**S**pearmen's **R**ank **C**orrelation **C**oefficient
ST ACF	**S**pace **T**ime **A**uto**C**orrelation **F**unction
STATS	**St**andardised **A**ccuracy and **T**ime **S**core
STGNN	**S**patial-**T**emporal **G**raph **N**eural **N**etwork
StoS	**S**equence **to S**equence learning
SVM	**S**upport **V**ector **M**achine
SVR	**S**upport **V**ector **R**egression
TRANSIMS	**Tr**ansport **An**alysis **Sim**ulation
VAPE	**V**alue **A**verage **P**ercentage **E**rror
VA	**V**alence **A**rousal
VGRAN	**V**ariational **G**raph **R**ecurrent **A**ttention **N**eural **N**etworks
VISSIM	**V**erkehr **I**n **S**tädten - **Sim**ulations modell

VISTA	**V**isual **I**nteractive **S**ystem for **T**ransport **A**lorithms
YOLO	**Y**ou **O**nly **L**ook **O**nce

Chapter 1

Introduction

1.1 Motivation and Background

Traffic congestion is a critical issue for all developed countries. In 2019 traffic congestion costs the UK economy an estimated £6.9 billion, and is predicted to rise [1]. The long-term motor vehicle traffic trends show that this rise is mainly due to huge growth in the use of cars and taxis, as illustrated in Fig. 1.1.

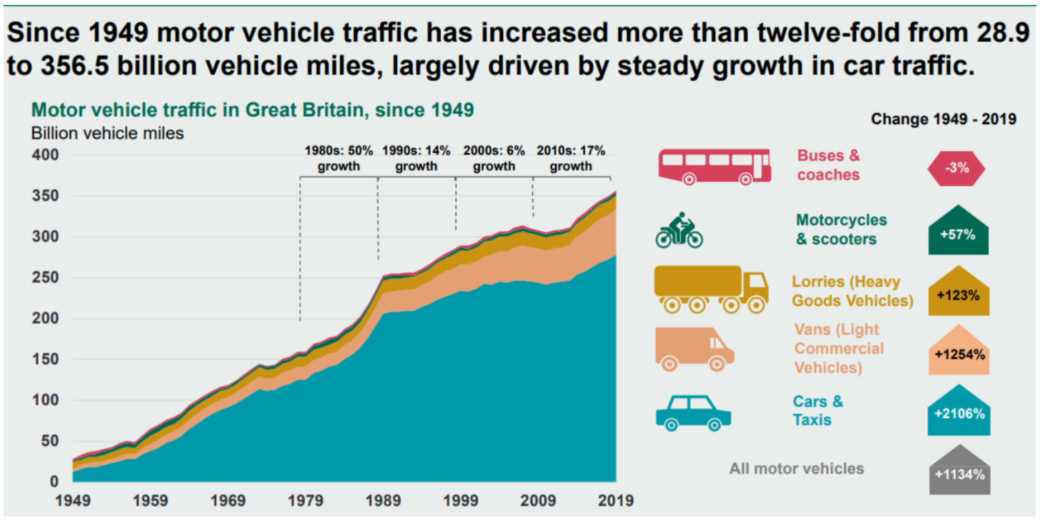

FIGURE 1.1: The long-term trends in motor vehicle traffic [2].

Congestion occurs when traffic demand exceeds the capacity of the road network (saturation point) [3]. A simplistic solution would be to build more or widen existing roads. However, space is a scarce commodity in most urbanised areas. Therefore, traffic managers have been given the arduous task of managing existing roads and networks to maintain or increase their capacity level.

Research in the last two decades has focused on Intelligent Transport Systems (ITS) development for the prediction and management of road traffic flow through advancements in data processing, mining, and computer hardware [4]. The main objective of these advancements is to move away from reactive methods of traffic management. These methods, such as CCTV monitoring and responding to reported incidents, can be costly in terms of operational and technical costs [5] as well as ineffective. Instead, proactive methods, which predict traffic congestion in real-time, can be used to prevent or alleviate future congestion through the use of congestion mitigation techniques [6].

Proactive methods for traffic management, such as short-term prediction models, can aid not only traffic controllers to make more informed decisions but also drivers by providing an early warning system. The key to an effective proactive method is a model that produces timely and accurate predictions. However, despite extensive research, an accurate and reliable proactive method is still not available [7]. Therefore, continued research into short-term traffic prediction models is vital.

Road traffic flow prediction methods can be broadly split into two categories; 1) traditional statistical methods, and 2) machine learning methods. Although, overlap between the categories does exist.

Traditional statistical methods, such as, but are not limited to, Auto-Regressive Integrated Moving Average model (ARIMA) [8], Seasonal Auto-Regressive Integrated Moving Average model (SARIMA) [8], Autoregressive Conditional Heteroskedasticity (ARCH) [9], and Generalised Autoregressive Conditional Heteroskedasticity (GARCH) [10] have all been used for road traffic prediction.

ARIMA, has produced promising results [11] [12][13][14], along with its seasonal variation SARIMA [15][16][17]. However, the model's simplistic nature means it is badly affected by outliers, and therefore, unable to cope with abnormal/non-recurrent (volatile) road traffic flow data. Some researchers have tried to overcome this limitation by creating hybrid models [18][19][20]. However, the model still suffers from a critical flaw, it assumes that traffic flow data is homoscedastic when it is generally accepted that road traffic flow is heteroscedastic. ARCH and GARCH models improve upon ARIMA and SARIMA models by assuming that road traffic flow data is heteroscedastic. Therefore, it can capture the volatility present in the road traffic flow data. The limited researchers that have used ARCH [9] and GARCH models [10] to predict road traffic flow [21] have produced good results, however, there is still room for improvement. ARCH and GARCH models, and their multiple variations, assume implied deterministic volatility based on historical road traffic flow and conditional variances. In reality, this is not true.

Machine learning methods that have been used for road traffic flow prediction include, but are not limited to, K-Nearest Neighbours (KNN) [22], Support Vector Regression (SVR) [23], and Artificial Neural Networks (ANNs) [24]. KNN models [22] are able to cope with volatile road traffic data [7][5], and therefore, many researchers have adapted KNN models for road traffic prediction [25][6]. However, the KNN model is not without issues, it requires large memory for storing the entire training dataset. Therefore, not suitable for the prediction of road traffic flow on a road network. SVR [23], is an extension of a classifier model, Support Vector Machine (SVM) [26]. Despite efforts to improve the time taken during the learning process [27][28][29], ultimately the SVR model does not work well with large high-dimensional datasets. Calculating the distances between each data point would be computationally very heavy, and this is only complicated further for high dimensional data. Therefore, SVR is not suitable for road traffic flow on a road network.

The use of ANNs [24] for road traffic flow prediction has flourished in the last two decades due to their ability to handle and predict non-linear volatile data and advancements in computing power. Various architectures have been explored [30], and predominately recurrent neural network (RNN) structures have been used [31]. These architectures include a Jordan Sequential Neural Network (JSNN) [32], a long short-term memory (LSTM) [33], and notably, a gated recurrent unit [34]. One main limitation of RNN models is they tend to focus on a small temporal dataset from one geographical site, ignoring spatial relationships within a road network. One direction that researchers are exploring to overcome this is convolutional neural networks (CNNs) [35]. A CNN is an ANN that uses the geographical proximity of its input data points to add a geospatial dimension, making them ideal for adaptation for the prediction of road traffic flow on a road network. Despite some promising research in [36][37], the application of CNNs for road traffic is limited and still in its infancy.

Therefore, there are still many challenges remaining for the short-term prediction of road traffic flow on a road network. The main issue with the ANN models mentioned above is there was no consensus on what architectural structure was superior for road traffic flow prediction. Furthermore, it identified one main flaw in the literature, the datasets used to test the models were inadequate. Most research focuses on a temporally small training and testing dataset, ranging from a few days to a few weeks [33][38], from one geographical location, and one data source/input features, that had been pre-cleaned to remove outliers for ease and speed of computation [38][39]. However, a prediction model can only be as good as its input data [40]. The magnitude of the temporal, spatial, and input features' diversity within the training data will determine and restrict what temporal, spatial, and input feature cycles and patterns can be learnt. Therefore, more research into the development of road traffic flow prediction models that can incorporate larger temporal datasets, from multiple locations on a road network, with diverse input features is needed. This challenge is not without its issues. Input features on a road network are dynamic, they change over time and space, and are highly correlated. Selecting the most discriminative input features for prediction is a difficult task. Therefore, more research in this area is also needed.

1.2 Aims and Objectives

This thesis aims to develop an accurate novel road traffic flow prediction model, with a particular focus on the short-term prediction of heterogeneous road traffic flow on an urbanised road network.

Objectives

1. To carry out a comprehensive literature review on road traffic flow prediction models (statistical, machine learning, and process-based) with applications to recurrent and non-recurrent road traffic flow.
2. Based on objective one, perform a benchmark evaluation of existing machine learning and deep learning prediction models for heterogeneous road traffic flow, with particular attention to the prediction model's stability to architectural change.
3. Based on objective two, develop an accurate prediction model that incorporates the historical temporal patterns of the heterogeneous road traffic flow to improve prediction accuracy.
4. To investigate input feature importance and develop a feature filter to improve the prediction accuracy.
5. To test and evaluate the models using real datasets from Transport for Greater Manchester.

1.3 Contributions

1. We have carried out a benchmark evaluation of existing machine learning models using a real dataset obtained from Transport for Greater Manchester. We investigated their prediction accuracy, time horizon sensitivity, and different input feature settings (different classes of vehicles) to understand how they can affect their prediction accuracy.
2. We have examined and compared existing deep recurrent neural networks (a basic recurrent, a long short-term memory, and a gated recurrent unit), using a novel performance metric we developed, based on a real dataset obtained from Transport for Greater Manchester to determine which is most suitable for the prediction of heterogeneous road traffic flow.
3. We have developed a novel online dynamic temporal context neural network framework that can dynamically determine how useful different temporal data segments are, and weights them accordingly for use in the regression model to improve prediction accuracy.
4. We have developed a novel dynamic exogenous feature filter that uses 'local windows' to filter input features in real-time to improve prediction accuracy.

1.4 The Structure of this Thesis

The rest of this paper is organised as follows: Chapter 2 presents a Literature Review for Road Traffic Flow Prediction; Chapter 3 is a comparative analysis of Machine Learning Based Approach for the Prediction of Road Traffic Flow; Chapter 4 is Evaluating the Performance of Deep Recurrent Neural Networks for the Prediction of Road Traffic Flow; Chapter 5 presents a Novel Online Dynamic Temporal Context Neural Network Framework for the Prediction of Road Traffic Flow; Chapter 6 introduces a Novel Dynamic Exogenous Feature Filter using Local Windows for Deep Learning-based Prediction of Road Traffic Flow; Chapter 7 provides a Conclusion.

Chapter 2

Literature Review of Road Traffic Flow Prediction

Chapter Summary

In the previous chapter we outlined the current issues surrounding road traffic flow prediction. In this chapter we move on to define, in detail, the scope of the mathematical problem and its components. Section 2.1 presents the components, including time parameter, input features, seasonality, traffic data and data sources, and performance metrics.

We then review and discuss various prediction modelling approaches for their suitability for the prediction road traffic flow. In Section 2.2.1 we present and assess three different statistical prediction models and their common variations, Section 2.2.2 evaluates four different machine learning models, and lastly, Section 2.2.3 examines process-based models. All the prediction models are appraised on their accuracy, time horizon, timestep, prediction scale, road structure, and input data used, and a conclusion is produced in Section 2.3.

2.1 The Mathematical Problem of Road Traffic Flow Prediction

Prior to performing a literature review first we define in general the mathematical problem of road traffic flow prediction. Equation 2.1 expresses the simplified problem and its components, when the conditions of the road network have been predefined.

The Mathematical Problem of Road Traffic Flow Prediction

$$\hat{x}_{t+1} = f(x_t, p_t, x_{t-1}, p_{t-1}, x_{t-2}, p_{t-2}, ..., x_{t-n}, p_{t-n}) \tag{2.1}$$

Where x is the traffic state, at current time t, p is the parameters affecting the traffic state, n is the size of the sample, $\hat{x}$ is the prediction of the traffic state, and $f(x)$ is the model applied to the historic traffic state data and its parameters to produce a prediction. Each component is now defined in more detail before exploring the different prediction models.

2.1.1 Time Parameters

A timestep (δt) is the interval of time between one point to the next ($\delta t = t_1 - t_0$) in the historical data and prediction. Past researchers have used a number of different timesteps in their predictions, however, most papers only predict one timestep ahead [6][41][42][28]; very few use multiple timestep prediction [25]. This is due to prediction accuracy becoming more volatile the more timesteps a model tries to predict.

A time horizon is the magnitude of the interval in time from one point to the next ($\Delta t = t_1 - t_0$) in the historical data and prediction. There is currently no universal definition of what magnitude time horizon is classed as short or long-term for the prediction of road traffic flow. Consequently, research papers have used a variety of different time horizons with varying success, ranging from 1 to 90 minutes [43], which have been labelled as short-term. However, a review of past papers (including, but not limited to those referenced in Table 2.3) appears to show that the most common time horizon used for short-term road traffic prediction is five minutes. Therefore, when referring to short-term traffic prediction in this research, it will be a time horizon of five minutes unless specified otherwise.

2.1.2 Input Features

Numerous input features (p) have been used in prediction models to improve prediction accuracy. These include, but are not limited to, speed, volume, density, state, occupancy, traffic flow, OD matrices, weather, truck flow, headway, and holiday/incident. A summary of the input features along with a count of the published papers they appear in can be seen in Figure 2.1, where the X axis represents the number of publications and the Y axis represents the input feature used in the publications. Figure 2.1 also shows the most commonly used input features are traffic speed, volume and flow. An explicit definition of volume, density, state, occupancy, traffic flow, and headway can be found in the Glossary.

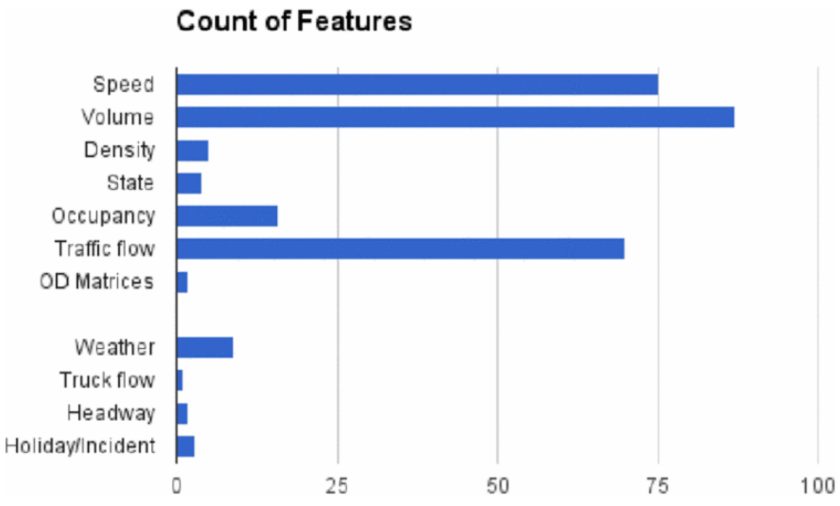

FIGURE 2.1: A count of the input features (p) used in published papers up to 2017 [44]

2.1.3 Seasonality

Equation 2.1 defines the general mathematical problem of road traffic flow prediction. One special case variation would be the inclusion of seasonality, as shown in equation 2.2.

$$\hat{x}_{t+1} = f_1(Bx) + \ldots + f_l(B^l x) + f_s(B^{si} x) \tag{2.2}$$

where $B^a x = x_t - a$, l is a non-seasonal term, s is a seasonal term, $l < s < \infty$, and $i = 1, 2, 3, \ldots n$.

Seasonality is a regular and predictable time series pattern within the historical road traffic flow data (x) that affects the two main constraints affecting traffic flow; 1) traffic demand, and 2) road capacity.

Road traffic flow exhibits strong seasonality patterns, such as short-term hourly and daily patterns [45][33][38] as well as weekly patterns connected to the working week [16]. Research has even demonstrated the importance of long-term seasonality patterns such as monthly and yearly [46]. Therefore, seasonality is a significant and well-studied factor that can aid road traffic flow prediction [45][47][48][49].

2.1.4 Traffic Data and Data Sources

Traffic data is classed as non-linear and complex [50] and is a fundamental part of any prediction model. It must be accurate, reliable, and complete to obtain a successful prediction [51]. Therefore, data can be pre-cleaned to remove outliers. Outliers in historical data, such as traffic jams or road closures, can skew data thus, negatively influencing prediction values for numerous iterations. Many researchers have devised methods to pre-clean noisy traffic data [52][53][54]. Reactive methods, such as the Box-Cox Transformations (TBATS) [52] where the geometric mean and the lambda (a key parameter of the transformations) that produces the lowest sum of squared error is used to determine how best to transform the data into homoscedastic data, can be used. Other more simplistic pre-cleaning methods can be used to insert values where data is missing, such as the average of the neighbouring values.

Some models are labelled as *robust* [55][56] denoting that they can predict and handle normal and abnormal traffic data. Based on existing work and research, traffic data that contains non-recurring traffic congestion or outliers is classified as atypical and thus, *abnormal*. Therefore, traffic data which does not contain non-recurring traffic congestion is classified as *normal*.

It should also be noted that the prediction of non-recurrent traffic events is impossible, except in special cases like planned public events or roadworks. However, some models can handle the dynamic data better during its occurrence and make reasonable predictions, hence, they are classed as robust.

Traffic Data Sources

Input feature data is obtained through three different sources; private, public, or simulated if no data is available. Suhas, Kalyan, Katti, *et al.* [44] analysed publications on road traffic predictions, through web scraping, to determine the most commonly used. The results can be found in Table 2.1 which shows that the majority of publications used privately sourced data.

Data Source	**Examples**	**Percentage**
Private	Regional Traffic Management Centres [28]	66.2
Public	PeMS, MIDAS [57], Bing[58] , and Twitter [59]	27.2
Simulated	SUMO [60] and AIMSUN [61]	6.6

TABLE 2.1: Sources of data used in road traffic prediction models up to 2017 [44]

2.1.5 Performance Metrics

To assess the accuracy of the prediction model, a performance metric must be employed. There is no standard performance metric used in road traffic prediction, thus, making the comparison of different papers and models difficult. A summary of the performance metrics, along with a count of the published papers they appear in can be seen in Figure 2.2. The X axis represents the number of publications the performance metric was found in, and the Y axis represents the performance metric used in the publications.

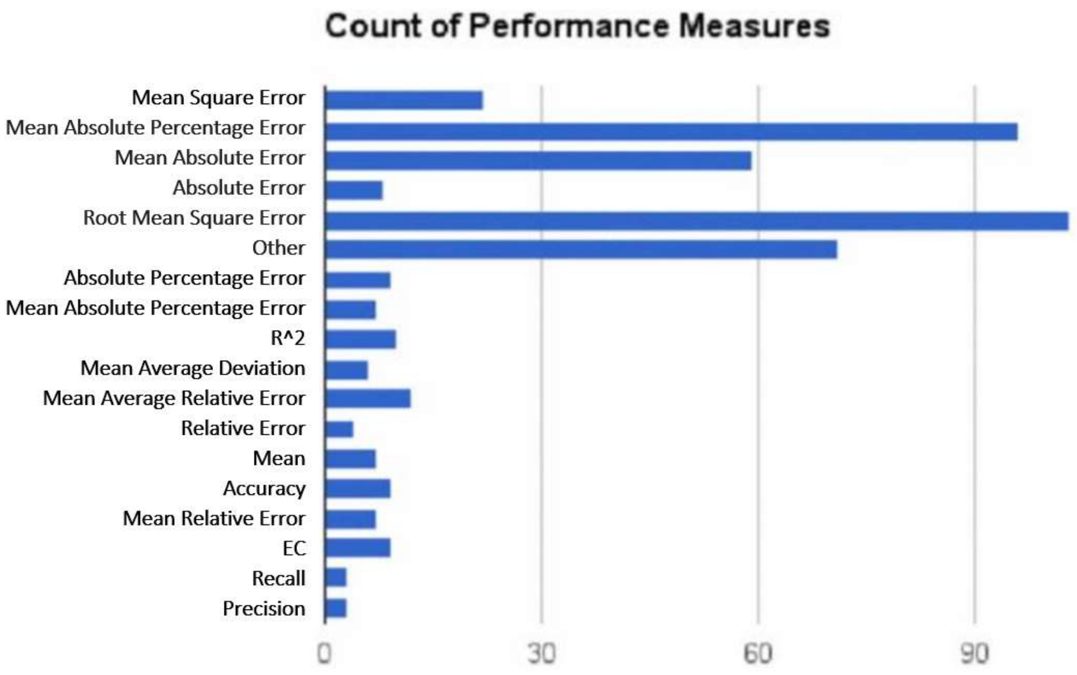

FIGURE 2.2: A count of performance metrics used in published road traffic prediction models up to 2017 [44]

The most commonly used performance metric is the *root mean square error* (RMSE), as shown in Equation 2.3. The RMSE measures the average deviation between the predicted value and the actual value. It only evaluates the accuracy of a model. No consideration is given to the training time, or the trade-off between accuracy and training time.

$$RMSE = \sqrt{\frac{\sum_{t=1}^{n}(F_t - A_t)^2}{n}} \tag{2.3}$$

F_t is the predicted value and A is the actual value at time t, and n is the number of timesteps predicted. During this research, the RMSE will be used to assess the prediction models' accuracy for ease of comparison with other research.

2.2 Traffic Prediction Models

Existing research into road traffic prediction can be broadly split into three different approaches; data-driven models (consisting of statistical and machine learning models) and process-based models, as shown in Figure 5.1, although hybrids do exist. Therefore, each approach is considered in turn.

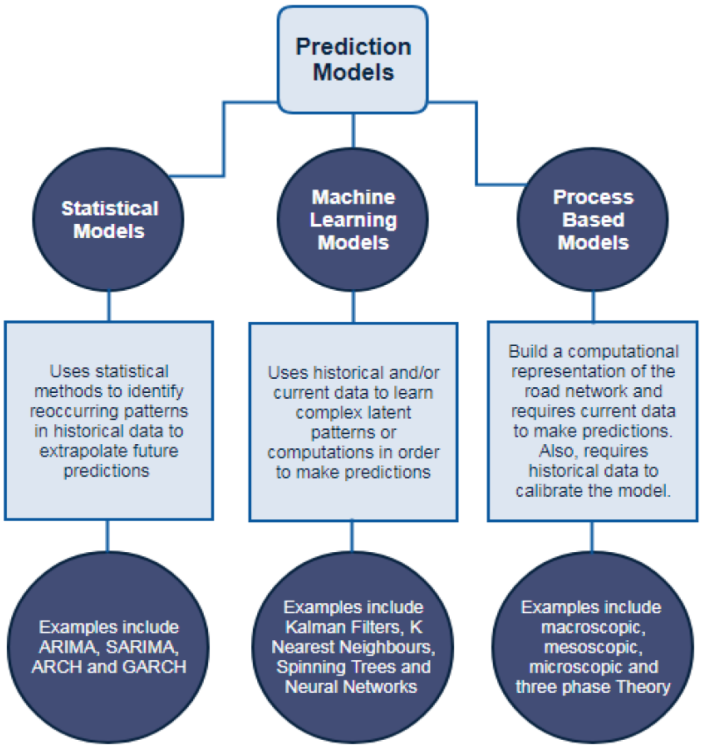
Prediction Models
Statistical Models
Machine Learning Models
Process Based Models
Uses statistical methods to identify reoccurring patterns in historical data to extrapolate future predictions
Uses historical and/or current data to learn complex latent patterns or computations in order to make predictions
Build a computational representation of the road network and requires current data to make predictions. Also, requires historical data to calibrate the model.
Examples include ARIMA, SARIMA, ARCH and GARCH
Examples include Kalman Filters, K Nearest Neighbours, Spinning Trees and Neural Networks
Examples include macroscopic, mesoscopic, microscopic and three phase Theory

FIGURE 2.3: Different approaches used for road traffic modelling

Within each approach there are numerous models. Thus, to determine which models should be explored, we perform a review of the most popular models . A review of published papers on traffic prediction models by Suhas, Kalyan, Katti, *et al.*, stating the percentage of papers that contained them, can be seen in Table 2.2. We explore the six most popular models, Artificial Neural Networks (ANNS) [24], deep learning, statistical, hybrid, Support Vector Machine/Support Vector Regression (SVM/SVR) [23], and K-Nearest Neighbours [62] Furthermore, we also review unpopular models, such as Grey Systems Forecasting (GSF) [63] and Kalman Filters (KF) [64], to access their suitability. The literature review assesses papers on their time horizon, timestep (presumed single if not otherwise stated), prediction scale (presumed single location point unless otherwise stated) and accuracy of results. The literature review begins with statistical models, the most popular class of all.

Prediction Techniques	Precentage of Studies
Neural Network	21.8
Statistical	30.7
Ensemble/Hybrid/Combined	19.6
Bayesian	3.6
Fuzzy	1.8
Deept Learning	1.8
SVM	5.8
SVR	4.0
ELM	1.3
KNN	4.0
HMM	1.3
Kalman Filter	2.2
Stochastic Process	1.3
Random Forest	0.9

TABLE 2.2: A comparison of the percentage of studies carried out using predictive techniques for Intelligent Transport Systems [44]

2.2.1 Traffic Prediction Models Based on Statistical Approaches

Statistical models are the predominately used method for short-term prediction (see Table 2.2). This is due to their simplistic structure and ability to produce results within an acceptable tolerable error level [5]. However, as they do not represent the road network and merely analyse the historical road traffic data, they can only predict congestion based on previous 'trends'. Thus, they are limited by the data available. Their performance depends on how closely the data represents the scenario under consideration. Below we discuss in more detail the most common statistical models used for prediction, Autoregressive Integrated Moving Average/Seasonal Autoregressive Integrated Moving Average, Autoregressive Conditional Heteroskedasticity/Generalised Autoregressive Conditional Heteroskedasticity, and Grey Series Forecasting.

1. ARIMA and SARIMA Based Traffic Prediction Modelling

In time series analysis, the most commonly used model for road traffic prediction is Auto-Regressive Integrated Moving Average model (ARIMA) [8][17]. It is a stochastic statistical parametric model that attempts to identify, linear and non-linear, traffic patterns in historic traffic time series data at a forecast location (arterial road or freeway) to predict future short-term traffic values.

Given time series x_t, the ARIMA model (p, d, q) is given as

$$\Theta_p(B)(1-B)^d(X_t-\mu) = \theta_q(B)e_t \tag{2.4}$$

where B is the back-shift operator given by $BX_t = X_{t-1}$, $\Theta_p(B)$ is the auto-regressive operator of order p, μ is the mean of series data, $\theta_q(B)$ is the moving average operator of order q, e_t is the random error (also known as white noise), and p, d, and q are the non-integer order of differencing [65].

ARIMA is used due to its simplicity and ability to produce computationally efficient predictions [11]. However, the model is limited by its reliance on historical data. It can not predict non-reoccurring traffic congestion and any outliers (such as traffic accidents or debris on the road) in historical data will also skew numerous successive future predictions. Therefore, it is not suitable for volatile or dynamic data. Furthermore, the model assumes that traffic data is homoscedastic. This means that it assumes the variance of the data is the same at any given time. This is not true since some periods during the day or week will be riskier and therefore, more volatile than others. Thus, the standard deviation from the mean during these periods will be higher than during the rest of the data. ARIMA does not and can not take this into account. Therefore, it is also unable to predict or cope with abnormal traffic conditions such as road traffic accidents. Nevertheless, ARIMA is suitable for locations where traffic flow is consistent and does not suffer from dynamic changes.

Two well-cited studies include Levin and Tsao [66] and Hamed, Al-Masaeid, and Said [67]. Levin and Tsao [66]. They determined that the ARIMA model (0, 1, 1) was most statistically significant at forecasting both volume and occupancy when compared to other ARIMA model variations and a 60 second time interval was most effective. Furthermore, Hamed, Al-Masaeid, and Said [67] also improved upon Levin and Tsao [66] research by making the model more computationally ‘tractable’. The calculations only needed to store the last forecast error and current data point making it computationally more efficient. However, the studies both only considered a small sample so it is questionable that generalisations can be made.

Despite some researchers successfully producing predictions within a tolerable error level [65][19], research has concentrated on improving the ARIMA model further to allow for abnormal traffic data. Hybrid models, combine the rigidity of the time series model with a non-parametric model, to improve its ability to cope with complex non-linear traffic flow data [11][49][19][16] and adding stochastic parameters such as accident and weather data [68][69] can improve the accuracy of the prediction.

Seasonal Auto-Regressive Integrated Moving Averages Model

Traffic data contains seasonality [70], in particular, urban areas tend to display a weekly pattern which is linked to the working week [16]. Therefore, it is advantageous to use a (SARIMA) model [8] to include the seasonal periodicity embedded in the time series data as a multiplicative model [15][21]. See Equation 2.5.

Given time series x_t, the SARIMA model $(p, d, q)(P, D, Q)s$ is given as

$$\Theta_p(B)\Theta_P(B^s)(1-B)^d(1-B)_s^D(X_t) = \theta_q(B)\theta_Q(B^s)e_t \tag{2.5}$$

where B is the backshift operator given by $BX_t = X_{t-1}$, $\Theta_p(B)$ is the auto-regressive operator of order p, μ is the mean of series data, $\theta_q(B)$ is the moving average operator of order q, e_t is the random error (also known as white noise), and p, d, and q are the non-seasonal order of differencing, P, D and Q is the seasonal order of differencing, and s is the seasonal factor, the timespan of the repeating seasonal pattern[16].

Lippi, Bertini, and Frasconi [71] used 4 months traffic flow data with 15 minutes time intervals from 16 traffic loop detectors using a daily seasonal factor, S of 96. They compared various SARIMA models and hybrids including Kalman filters and Artificial Neural Networks (ANN) and concluded that a seasonal factor is a key feature to achieving accurate predictions, although ultimately they concluded that hybrid models were more effective.

This is due to the SARIMA model suffering from the same issues as the ARIMA model. It too assumes the traffic input data is homoscedastic. However, despite the obvious flaws, it is generally accepted that it is the most precise and widely used prediction model in its field for the prediction of road traffic flow [72]. Nevertheless, some researchers have argued that taking into account seasonal effects do not significantly impact the prediction when compared with simpler models [73]. Furthermore, their research has been criticised for using abnormal traffic data [16], which the ARIMA model is not suitable for. A model that assumes the traffic data is heteroskedastic is now assessed.

2. ARCH and GARCH Model

Auto-Regressive Conditional Heteroskedasticity (ARCH) [9] model is an improvement upon ARIMA and SARIMA models as it assumes that the input traffic data is heteroscedastic and therefore, can handle volatile traffic data. It can do this through the addition of a prediction for each error term by measuring the volatility in a time series and predicting future volatilities through the addition of weights (manually weighted). The next data point's variance is modelled as a function of the previous independent and dependent variables, a weighted average of the previous data points' variances (see Equation 2.6).

Given time series y_t the ARCH model is given as

$$y_t|\psi_{t-1} \sim N(x_t\beta, h_t) \tag{2.6}$$

variance, h_t, is given as

$$h_t = h(\psi_{t-1}, \alpha) \tag{2.7}$$

the error term, ε_t is given as

$$\varepsilon_t = c + u_t + \sum_{n=1}^{r} \alpha_n \varepsilon_{t-i}^2 \tag{2.8}$$

Where x and α are a vector of unknown parameters, $x_t\beta$ is the mean of y_t which is assumed to be a linear combination of the lagged independent and dependent variables, α is the positive efficient of the lagged sample variance, ψ is the dataset, u_t is the white noise, c is a constant, and t is time [9].

To overcome the subjective nature of the weights and simplify the model further an adaptation of the ARCH model, the Model (GARCH) [21], has been used in road traffic prediction.

Generalised Autoregressive Conditional Heteroskedasticity

The model [21] uses a weighted average of the past squared residuals. It has declining weight, which tends to zero although, never reaching zero (See Equation 2.9). It is simplistic and easy to predict the conditional variances. Given that the conditional variance, h_t, is given as

$$h_t = c + \sum_{n=1}^{q} \alpha_n \varepsilon_{t-n}^2 \tag{2.9}$$

future predicted values for time series x_t are given as

$$h_t = \sum_{m=1}^{s} \delta_m h_t - m + \sum_{n=1}^{r} \alpha_n \epsilon_{t-n}^2 \tag{2.10}$$

where p is the order of the auto-regressive element of the equation and $p \geq 0$, q is the order of the moving average element of the equation and $q \geq 0$, α_0 is the positive coefficient, α_i is the positive coefficient of the lagged sample variance ε_{t-i}^2, and β_i is the non-negative coefficient of the lagged conditional variance h_{t-i} [21]. When traffic flow data is suddenly volatile due to an accident or debris on the road it would seem logical that the next prediction should be volatile too; accidents and debris can not clear instantaneously. It will take time, thus, a series of data points for the road issue to be resolved and to return to normal driving conditions. Due to the weighting, the model can handle volatility by showing a preference for the most recent variances. Furthermore, due to the weighting of the past data variances tending to zero, any volatility in the data such as accidents or major events will not skew future predictions. The model predictions normalise themselves while the road traffic flow returns to normal driving conditions.

Research using ARCH and GARCH models for road traffic congestion is limited [74]. However, Gavirangaswamy, Gupta, Gupta, *et al.* [21] used 3.5 years of traffic flow and volume data at one hour intervals to compare ARIMA, SARIMA, and ARIMA-GARCH model (see Equation 2.11).

$$x_t = \sum_{i=1}^{p} \alpha_i x_{t-i} + \sum_{j=1}^{q} d_j \epsilon_{t-j} + k + \sum_{m=1}^{s} \delta_m h_t - m + \sum_{n=1}^{r} \alpha_n \epsilon_{t-n}^2 \tag{2.11}$$

Note that the first 2 terms in Equation 2.11 refer to the ARIMA model and the last 2 terms refer to the GARCH model.

Gavirangaswamy, Gupta, Gupta, *et al.* [21] concluded that the ARIMA-GARCH model was the most stable and gave the most accurate results in a one-step prediction. This was due to the GARCH model's ability to encompass the short and long-range dependencies of the historical data [21]. This has practical application for ITS. A model that can make predictions within a reasonably tolerable level on various road types and can predict normal and abnormal traffic congestion would be of great interest. However, the data used was one-hour intervals. This does not appear to be the standard time interval for short-term traffic prediction. Would the GARCH model produce such stable results if the time intervals were much smaller? Using larger time intervals has a 'smoothing' effect on the data. Thus, making the data less volatile and easier to work with. Reproducing the experiment using smaller time intervals would also have more practical benefits for traffic planners. As traffic data is so volatile, traffic states can rapidly deteriorate. Therefore, Traffic Planners need to know the future traffic state in the next five to ten minutes to be able to successfully mitigate future traffic congestion.

All previous models mentioned (ARIMA, SARIMA, ARCH and GARCH) can be classed as time series and regressive analysis. We will now explore a non-functional model.

3. Grey Series Forecasting Model

Grey Series Forecasting (GSF) [63] is a non-functional and non-linear model which does not rely on the classic statistical method of regression analysis and is based on the principles of Grey System Theory (GST) [75] developed by Deng. GSF models can be used to analyse latent and complex patterns in traffic networks using only a small amount of data [76], due to a key aspect of GSF is its assumption that latent patterns can be identified using limited data (at least 4 data points are needed), dependent on the model's predictor window size. Thus, negating the need to store large volumes of data [63]. Instead, the model can dynamically update the parameters of the model with the traffic flow data [7], thus, reducing its reliance on historical patterns. This characteristic is advantageous in traffic modelling as it enables the model to predict road traffic congestion under abnormal traffic conditions and does not allow historic outliers to skew future predictions.

The GSF model works in three steps. See Figure 2.4.

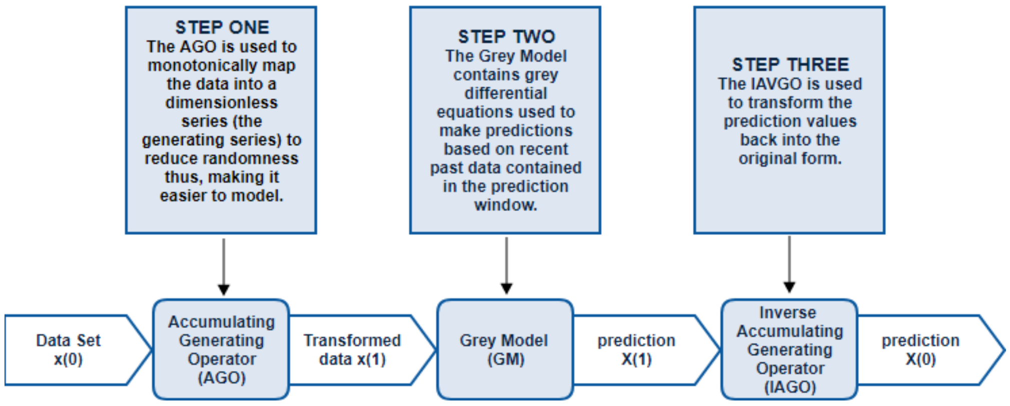
STEP ONE
The AGO is used to monotonically map the data into a dimensionless series (the generating series) to reduce randomness thus, making it easier to model.
STEP TWO
The Grey Model contains grey differential equations used to the make predictions based on recent past data contained in the prediction window.
STEP THREE
The IAVGO is used to transform the prediction values back into the original form.
Data Set x(0)
Accumulating Generating Operator (AGO)
Transformed data x(1)
Grey Model (GM)
prediction X(1)
Inverse Accumulating Generating Operator (IAGO)
prediction X(0)

FIGURE 2.4: A Grey Series forecasting model [63]

Simplistically this can represented as

$$AGO \cdot GM \cdot IAGO : x^{(0)} \to \hat{x}^{(0)}(\xi) \tag{2.12}$$

where the superscript 0 denotes the original raw data, $\hat{x}^{(0)}(\xi)$ denotes the calculated prediction value at time point ξ. More formally, given time series $x^{(0)}(k)$ (where $k = 1, 2, ..., n{:}n \geq 4$) a $GM(n, m)$ (where n is the order of the differential equation and m is the number of variables) can be defined as

$$x^{(1)}(k) = \sum_{i=0}^{k} x^{(0)}(i) \tag{2.13}$$

where superscript 1 indicates the data transformed by the AGO. Therefore, $GM(n, 1)$ can be defined as

$$x^{(0)}(k) + az^{(1)}(k) = b \tag{2.14}$$

where a is the developing coefficient and b is the Grey input. Both a and b are found by

$$\begin{bmatrix} a \\ b \end{bmatrix} = \hat{a} = (B^T B)^{-1} B^T y \tag{2.15}$$

In 2.15, $z^{(1)}$ represents the mean value of the transformed data, c is a coefficient subject to $0 \leq c \leq 1$ but typically fixed at 0.5 [63], B is given as

$$B = \begin{bmatrix} -z_{(1)}^{(1)}(2) & x_{(2)}^{(1)}(2) & x_{(3)}^{(1)}(2) & \cdots & x_{(n)}^{(1)}(2) \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ -z_{(1)}^{(1)}(n) & x_{(2)}^{(1)}(n) & x_{(3)}^{(1)}(n) & \cdots & x_{(n)}^{(1)}(n) \end{bmatrix} \tag{2.16}$$

and y is given as

$$y = \begin{bmatrix} x_{(1)}^{(0)}(2) \\ x_{(1)}^{(0)}(3) \\ \dots \\ x_{(1)}^{(0)}(n) \end{bmatrix} \tag{2.17}$$

Therefore, the Grey differential equation is defined as

$$\frac{dx^{(1)}(t)}{dt} + ax^{(1)}(t) = b \tag{2.18}$$

which can be solved for time k using

$$\hat{x}^{(1)}(k+1) = \left(x^{(0)}(1) - \frac{b}{a}\right)e^{-ak}(1-e^{a}) \tag{2.19}$$

[63]

Guo [7] compared a grey model (GM(1,1)) to a SARIMA$(1,0,1)(0,1,1)_{96}$ using traffic flow data of unspecified amount and concluded that the GM(1,1) model performed better than the SARIMA for both normal and abnormal traffic conditions. Particularly abnormal traffic where the mean absolute percentage error (MAPE) fell from 37.47% to 22.97% [7]. Guo attributed this to the model's ability to detect and react to traffic incidents [7] due to its small prediction window (the limited historical data used) and its dynamically changing parameters. However, Guo failed to investigate the model further. Investigating how the GM(1,1) compares with other variations and adaptations of the GSF would be of interest. Kayacana, Ulutas, and Kaynaka [42] used GSF and its adaptations on two years of noisy financial data. The adaptations used were EFGM (a GM with Fourier Series), TFGM (a GM with triangular fuzzy-number), GVM (a Grey Verhulst Model), EFGVM (a GVM with Fourier Series), and TFGVM (a GVM with triangular fuzzy-number). He concluded that the adaptations performed better, than the standard GM(1,1). Specifically, the GM(1,1) adapted to use Fourier series was the best model in terms of fitting and forecasting the data. To the best of our knowledge, road traffic prediction has not been explored with GM adaptations. Although this is not explicitly stated in Guo research, it may be due to one of the model's biggest flaws; its need for faultless data. In real-life traffic data collection, this is not always possible. This may be a key reason why research into road traffic prediction using GSF is very limited.

2.2.2 Traffic Prediction Models Based on Machine Learning

Machine learning models are popular for road traffic prediction due to their ability to learn without being explicitly programmed and make predictions based on vast amounts of data. We now explore the most popular machine learning models (see Table 2.2).

1. K-Nearest Neighbour Based Traffic Prediction Models

K-Nearest Neighbour (KNN) [22][62] is a non-linear, non-parametric, lazy learning pattern recognition model used predominantly for classification as well as regression. It does not attempt to learn the complex latent relationships between the variables, instead, it predicts future traffic volumes by working on the assumption that observations that are near each other in the feature space are from the same class. Thus, classifying the current traffic state by the number of k neighbours and their weightings. The KNN prediction model works in four main steps. Firstly, historical data which is representative of the road network and large enough to deal with various scenarios is located. Secondly, a suitable distance matrix is determined. Thirdly, a suitable value of k is determined and the historical data is searched for k nearest neighbours using the distance function. Lastly, a prediction algorithm (weighting) is determined and based on the k nearest neighbours a prediction is made for the next time horizon [77]. See Figure 2.5.

An advantage of the model is it does not rely on periodicity in historical data, therefore, the model can adapt to dynamic changes [16][62]. It can handle and predict volatile traffic flow such as road traffic accidents or debris on the road. Furthermore, the model's simplicity allows it to be a computationally efficient model. However, the model's accuracy is determined by how closely the training data represents the road's volatility. The model is unable to predict previously unseen traffic congestion.

The three parameters of the model that can be changed to improve prediction are the distance function, the choice of 'k', and the prediction function. The first KNN model used in road traffic prediction was Davis and Nihan [62] who experimented with the choice of k. They used a small highway dataset to compare the three different KNN models (using different values of k) against an ARIMA model. It was concluded that the model was comparable but not significantly better than

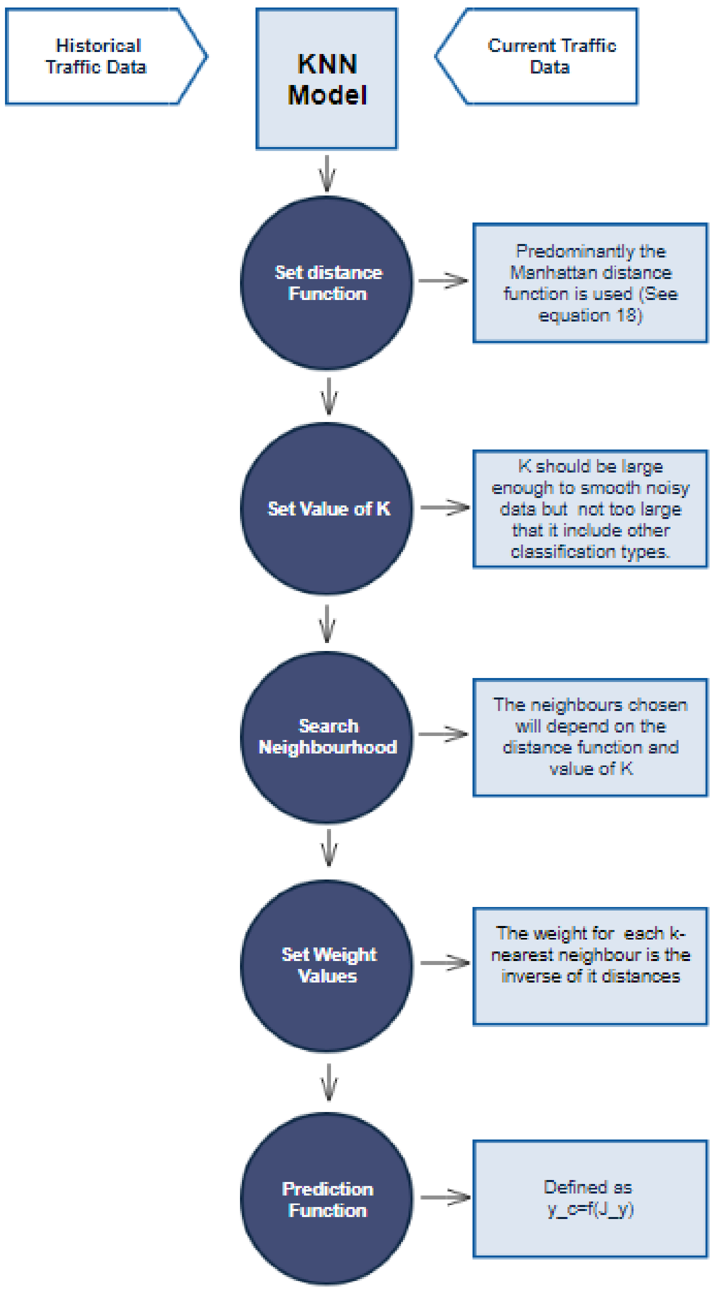
Historical Traffic Data
KNN Model
Current Traffic Data
Set distance Function
Predominantly the Manhattan distance function is used (See equation 18)
Set Value of K
K should be large enough to smooth noisy data but not too large that it include other classification types.
Search Neighbourhood
The neighbours chosen will depend on the distance function and value of K
Set Weight Values
The weight for each k-nearest neighbour is the inverse of it distances
Prediction Function
Defined as y_c=f(J_y)

FIGURE 2.5: A K-Nearest Neighbours model

the ARIMA model. However, Davis and Nihan determined this was most probably due to the small amount of historical data used for the KNN search. Many other researchers have had more success implementing a KNN model for road traffic prediction [77][50][78].

More recently, one parameter research has focused on is the distance function. It can be calculated in many ways, for example, the Chebychev, Hamming, or Spearman distance function. The most common are shown in Equations 2.20 to 2.22.

Euclidean

$$\sqrt{\sum_{i=1}^{k}(x_t - y_t)^2} \tag{2.20}$$

Cityblock/Manhattan

$$\sum_{i=1}^{k}|x_t - y_t| \tag{2.21}$$

Minikowski

$$\left(\sum_{i=1}^{k}(|x_t - y_t|)^q\right)^{\frac{1}{q}} \tag{2.22}$$

The most suitable distance function will depend on the type of data used in the model. For example, Euclidean Distance (see Equation 2.20) is used when all variables are the same type (all measurements in meters etc.). Whereas, for traffic data which includes different types of variables (speed, density, precipitation, etc.) the Manhattan Distance, also known as City Block Distance, is considered the most suitable (see Equation 2.21). Nevertheless, many researchers have sought to improve upon the standard distance function.

Cai, Wang, Lu, *et al.* [25] used a small dataset of 4 weeks (weekdays only) of speed and flow data (15 days of training and 5 days of testing) with a time horizon of 5 minutes to develop a spatiotemporal correlation KNN model. In a traditional time series KNN model, the data is marked by time. Alternatively, Cai, Wang, Lu, *et al.* also used the physical distance of the road segments in a spatiotemporal matrix in the calculation of the distance function which used a Euclidean distance function. The results showed that the addition of a spatial parameter improved the accuracy of the prediction not only for a single timestep but for multi-timesteps. However, taking the data into a spatiotemporal matrix increases its dimension and thus, increases its computational time and complexity. The model also used the Euclidean distance function stating that this was traditionally the most used. This may be true, due to most studies using a single input parameter such as traffic flow or volume, nevertheless, for traffic data which contains different types of variables (speed, distance, flow, etc) Cityblock/Manhattan Distance (Equation 2.21) would be most suitable.

The most commonly used weights in the KNN model are the inverse distance (Equation 3.1) and the attributes weights, though, many researchers have sort to improve upon these simplistic methods [27][79]. Sun, Cheng, Goswami, *et al.* [6] proposed a model that weighted the k nearest neighbours dynamically based on the traffic flow rate known as the weighted parameter tuples (WPT). The research used one year of highway data from Kunshi, Bejing (split 80% for training and 20% for testing) at five minute time horizons. The study concluded that dynamically weighted k values outperformed statically weighted k values. However, the model was never compared to other machine learning models such as Artificial Neural Networks. In conclusion, KNN is a well established non-parametric prediction model that requires no fitting to the road network and can cope with and predict recurrent and non-recurrent traffic congestion.

2. Kalman Filter Model Based Traffic Prediction Models

(KF) [64] was first developed by Kalman [64]. It is a stochastic parametric recursive model used for the prediction of linear dynamic data that is discrete in its time domain and contains uncertain parameters. However, the Extended Kalman Filter (EKF) is a common adaptation for non-linear data. Therefore, the model can cope with and predict volatile (unseen) traffic data [51] such as road traffic accidents and planned public events.

Kalman Gain

$$K_t = \frac{P_t' H^T}{H P_t' H^T + R_t} \tag{2.23}$$

Estimation Update

$$\hat{X}_t = \bar{X}_t + K_t (y_t - H_t \bar{X}_t) \tag{2.24}$$

Covariance Update

$$P_t = I - K_t H_t P_t \tag{2.25}$$

Prediction

$$\hat{X}_{t+1} = A \hat{X}_t \tag{2.26}$$

$$P_{t+1} = A P_t A^T + Q \tag{2.27}$$

It is modelled on Markov chains and linear operators. The model makes predictions using the estimated 'state'. It has 2 main steps. First, is the prediction step; in this step, the time is updated and the most recent state and error covariance (from the previous timestep) are projected forward to produce a predicted (priori) estimate for the current state. Next, is the update step; the measurements are updated and

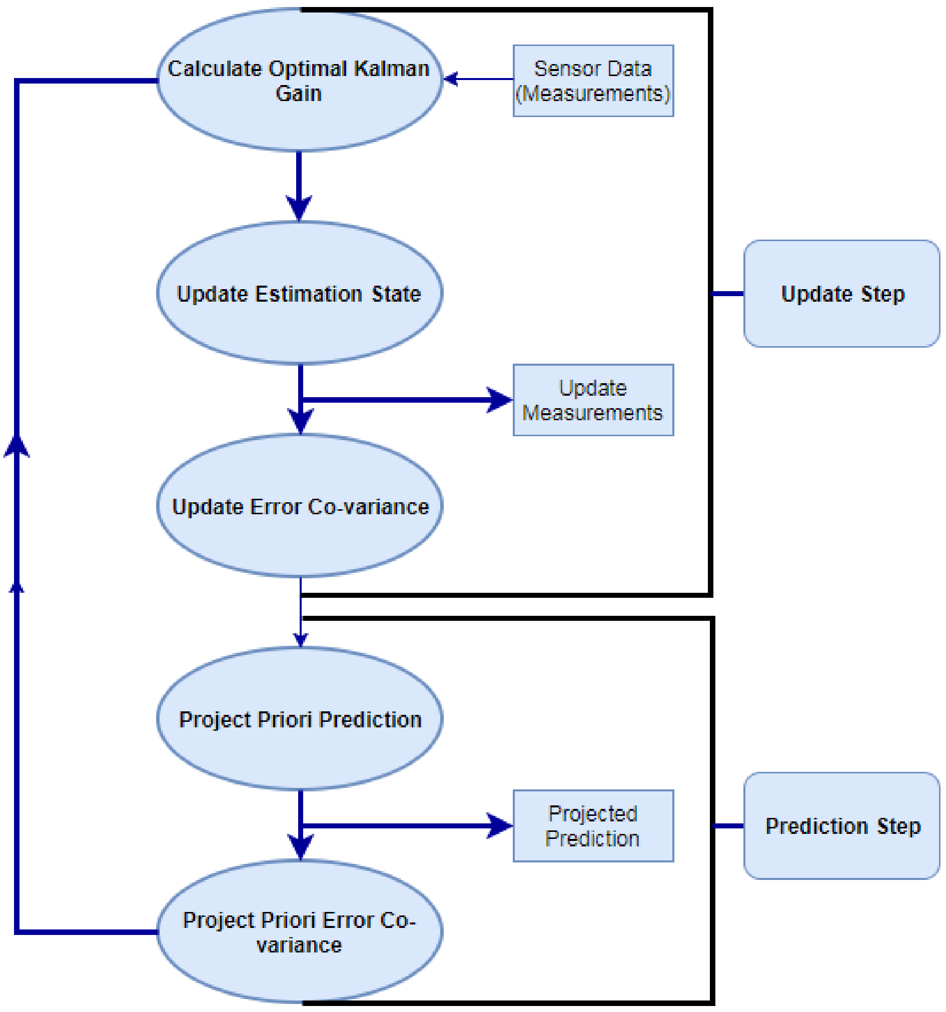
Calculate Optimal Kalman Gain
Sensor Data (Measurements)
Update Estimation State
Update Step
Update Measurements
Update Error Co-variance
Project Priori Prediction
Projected Prediction
Prediction Step
Project Priori Error Co-variance

FIGURE 2.6: A Kalman model

the estimated predicted state from step one is corrected using the newly observed data. This continues in a cycle. See Figure 2.6.

Therefore, the KF successively estimates the variables and their uncertainties. At each new data point, the current variables and their uncertainties (random noise errors) are updated using the newly observed data and weighted averages to estimate the future variables. This is achieved through two equations, the state equation and the observation equation.

Given time series x_t, the state equation can be defined as

$$x_t = F_{t-1}x_{t-1} + B_t u_t + w_{t-1} \tag{2.28}$$

F is transition/motion matrix which determines the transition from state $t-1$ to state t, B is a control input matrix for the parameters in vector u, u is a vector that contains any control inputs, and w is white noise process which is assumed to be Gaussian.

Given observations from the road traffic sensor site (RTSS), y_t, the observation equation can be given as

$$y_t = H_t X_t + N_t \tag{2.29}$$

where H is a matrix that maps the input state into the process state and N is the white noise measurement which is assumed to be Gaussian.

KF was first used in road traffic prediction by Gazis and Knapp [80]. Traffic speed and flow measurements were used at two RTSSs (an entrance and exit point) and it was found that the KF was able to predict the road traffic with great accuracy [81]. However, no explicit details on the traffic data are given and no comparisons to other models were made.

More recently, Xu, Wang, Jia, *et al.* [14] used four days of speed and volume measurement data three days for training and one day for testing) with a time horizon of two minutes from four different road segments in Beijing to compare ARIMA, KF, particle filter and a hybrid of the KF and ARIMA model. It concluded that the hybrid of the KF and ARIMA model was the most successful model for predicting the traffic state. However, the study failed to explore spatial-temporal correlations. These correlations have been shown to improve accuracy in other studies. Also, one criticism of the study that Xu, Wang, Jia, *et al.* acknowledge is the lack of updating in the optimal parameters. The optimal parameters are determined by the historical data, yet the current optimal parameters may be different to the historical ones.

Mir and Filali [41] improved the optimal parameters through the estimation of the process noise. One month of speed measurement data with five minute time horizon was used to develop a scalar adaptive KF model. The model attempts to minimise the difference between the actual speed measurement and its estimation. This is done by dynamically adjusting the process noise (Q in Equation 2.27) based on the current speed. The work compared the adaptive process noise model to a model with no process noise estimation, and KF models with and without measurement noise (N in Equation 2.29). It was concluded that due to 'on the spot' speed does not fluctuate greatly, the model was able to make reasonable predictions. However, the model was only tested on normal data. No abnormal traffic conditions such as sudden severe congestion.

In conclusion, the KF model has been proven to provide successful road traffic predictions [82]. It is computationally efficient due to only needing the previous data point to calculate the current estimate, therefore, it is suitable for calculations in real-time. However, the model assumes that all noise is Gaussian. Non-linear models may be more suitable.

3. Support Vector Regression Based Traffic Prediction Models

SVM [26] is a well known non-parametric supervised learning classifier used to predict discrete categorical labels. A well known adaptation of this model is Support Vector Regression (SVR) [23], developed by Scholkopf, Simard, Smola, *et al.* [23]. Through the inclusion of an extra slack variable for each training point, the model can predict continuous ordered variables.

The SVR model works in two steps, a classifier and a learning step. The classifier step works by taking the training data $((x_1, y_1), (x_2, y_2), ..., (x_n, y_n))$, which is contained within the input space (X), and mapping it through a mapping function (Φ) into a higher dimensional feature space (F). This is done so that a non-linear function $(f(x))$ can then be found (see Equation 2.30) with 'ϵ' precision. Therefore, all the training data is no more than ϵ deviations away from the target training data. This is done because it is assumed that a high dimensional linear regression is the equivalent to a low dimension non-linear regression [83]. More formally, this can be denoted as

$$f(x) = \langle w + \Phi x \rangle + b \tag{2.30}$$

where $\langle w + x \rangle$ denotes the dot product of X, w is a weight vector in the feature space (F), and b is the bias of the regression function.

Next is the learning step. In order to produce an accurate prediction, the optimal values for b and w must be found. For w, a small value that produces an error less than Φ deviations away from the actual training data is needed. This can be found using

$$\frac{1}{2} \parallel w \parallel^2 \text{ subject to } \begin{array}{l} y_i - \langle w, x_i \rangle - b \geq \epsilon \\ \langle w, x_i \rangle + b - y_i \geq \epsilon \end{array} \tag{2.31}$$

However, this equation produces a problem for use with road traffic prediction.

Due to traffic flow containing non-linear and complex characteristics [50] the constraints in Equation 2.31 will make the optimisation of w not feasible. Therefore, slack variables (ζ_i and ζ_i^*) to create a 'soft margin' for the ϵ deviations was developed so the model is able to handle non-linear data such as traffic flow data. Thus, the new value of w can be found as

$$\frac{1}{2} \parallel w \parallel^2 + C\sum_{i=1}^{n}(\zeta_i + \zeta_i^*) \text{ subject to } \begin{array}{c} y_i - \langle w, x_i \rangle - b \geq \epsilon + \zeta_i \\ \langle w, x_i \rangle + b - y_i \geq \epsilon + \zeta_i^* \\ \zeta_i, \zeta_i^* \geq 0 \end{array} \tag{2.32}$$

where C is a constant subject to $C > 0$ that determines the number of deviations larger than ϵ will be tolerated, and n is the number of data points.

Another adjustment of the SVR that affects road traffic congestion prediction is the replacement of the dot product ($\langle w, x_i \rangle$) in Equation 2.30. Instead of the dot product, a kernel function can be used. This negates the need for explicitly using the mapping function (Φ) to map the dot product into the higher dimensional feature space. There are various kernel functions used in SVR. The most common kernel functions are

Linear

$$k(x, y) = xy \tag{2.33}$$

Polynomial

$$k(x, y) = xy^t \tag{2.34}$$

Radial Basis Function

$$k(x, y) = exp\left(\frac{- \parallel x - y \parallel^2}{2\sigma^2}\right) \tag{2.35}$$

For road traffic congestion the most widely used kernel function is the RBF. This is due to its highly effective mapping of non-linear data [84] as it is a 'local' kernel function. Thus, it has a strong learning ability but a weak generalisation. Therefore, extreme historical variances in road traffic data will not negatively affect future predictions.

Hu, Yan, Liu, *et al.* [28] used three months of motorway data with a time horizon of 15 minutes to develop a Particle Swarm Optimisation (PSO) and SVR hybrid. The research focused on replacing the standard equation 2.32, which calculates the optimal parameters for the SVR, with a PSO. This is done to reduce the time taken in the learning step to aid real-time prediction. The results were compared to various other ML algorithms such as KNN and ANN. It was found that the results were comparable but not a significant improvement. However, the study failed to compare the computation time of the model against other ML models, which appeared to be the premise of the research. Only times for the PSO-SVR variations were compared. The study also failed to take into account any spatial parameters. It only considered traffic speed, flow and journey time.

Ling, Feng, Chen, *et al.* [29] improved upon the Hu, Yan, Liu, *et al.*'s work further by using a PSO that not only took into account historical data but also real-time data - an adaptive particle swarm optimisation (APSO). It also sought to improve the mapping function. As road traffic data is both random and non-linear [50], a 'multi-kernel' approach was developed. A hybrid of an RBF and a polynomial kernel was used. This was deemed more suitable for very dynamic road traffic data as it can utilise the Gaussian kernel's ability for local learning and the polynomial kernel's ability for global generalization. The results for the RBF, polynomial and hybrid kernel were MAPEs of 15.76, 16.52, and 10.57 respectively. The study was tested on both urban and freeway roads. However, although the study did use volatile rush hour data, no abnormal non-recurrent traffic congestion data was used. In theory, the use of a polynomial kernel should negatively affect the model's ability under these conditions. In conclusion, even though SVR is at its core a linear classifier it is still able to cope with and predict non-linear data [27], although its choice of the kernel is key.

4. Artificial Neural Networks Based Traffic Prediction Models

(ANN) [24] are non-parametric flexible prediction models that can model the complex latent non-linear relationships between the input data and the output data [20]. Therefore, ANNs have been applied to road traffic prediction due to their ability to handle and predict non-linear dynamic data [85]. First proposed by McCulloch and Pitts [24], the essence of the ANN is quite simplistic. It is made up of three basic layers, an input layer, a hidden layer, and an output layer. See Figure 2.7.

An ANN takes the training data $(x_1, x_2, ..., x_n)$ and inputs it into the top layer, the input nodes (nodes are also known as neurons). The input nodes are connected

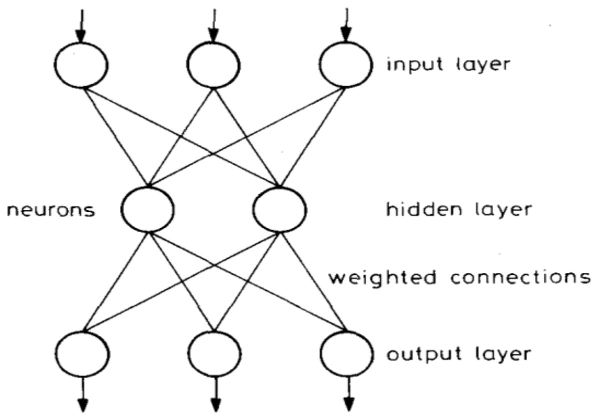

FIGURE 2.7: A three-layer feed-forward neural network [86]

through edges to the hidden layer nodes. Each incoming edge has a 'weight' ($w_1, w_2, ..., w_n$) (also known as biases) associated to it. The incoming data is multiplied by the weight and if the sum of all the weighted incoming data (as shown in Equation 2.36) meets a set threshold then the activation function is applied to the sum (Z) as shown in Equation 2.37 and the data is passed on through the edges to the next layer. At first, the weights and thresholds are set at random and they are continuously updated until the transformed data reaches the output layer resembling the output of the training data. An ANN with multiple hidden layers is known as deep learning. Once the network has been trained with the training data the test data is then passed through. This is known as a Feed Forward Neural Network (FFNN) as the data only moves in one direction. However, an ANN that contains some form of learning rule or rules (a learning assignment) and the data can be passed back to modify the weights and thresholds is known as a Back Propagation Neural Network (BPNN) **1990ProbabilisticNetworks**.

Transfer Function (Sum of the Node)

$$Z = \sum_{i=1}^{n} w_i x_i \tag{2.36}$$

Activation Function

$$Y = \varphi(Z) \tag{2.37}$$

Although BPNN can produce good prediction results, the results can be improved by extending the basic BPNN through different training procedures, different internal structures or mathematics, preprocessing the input data or including spatial and/or temporal patterns in the model [51][87].

More, Mugal, Rajgure, *et al.* [32] developed a model that improved the internal structure of standard ANN by using a Jordan Sequential Neural Network (JSNN). The optimal structure of an ANN is based on variances present in the historical data. However, the variances present in the historical data may vary from the variances in the test data, after all, traffic data is complex and stochastic [68], leading to inaccuracies in the prediction. To overcome this a JSNN adds 'context' to the prediction by allowing it to 'remember' the most recent past data (partially recurrent network). As shown in Figure 2.8, the hidden layer receives input from the current timestep (t) and context from the context layer from the previous timestep ($t - 1$).

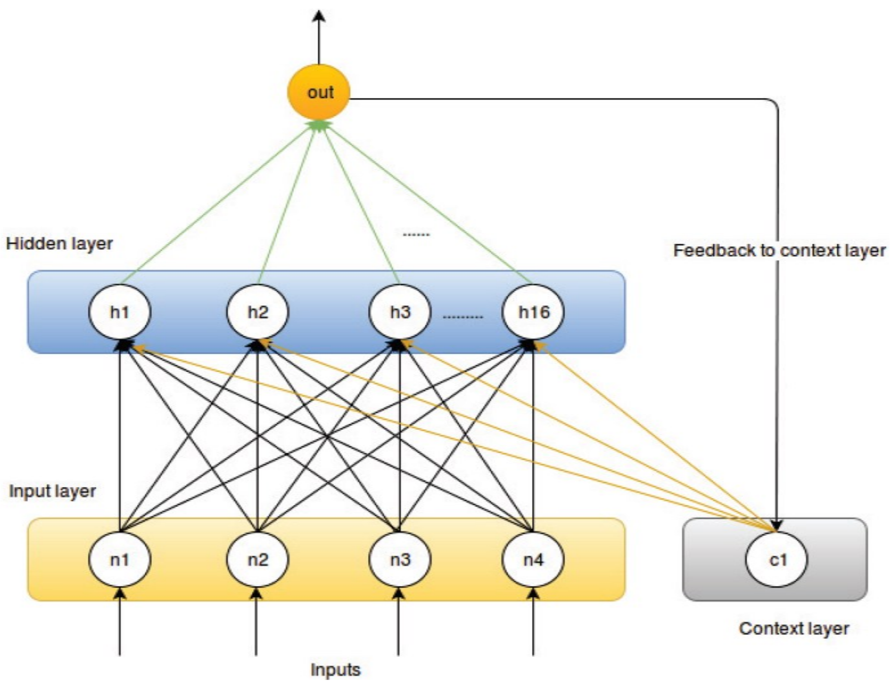

FIGURE 2.8: A Jordan sequential neural network [32]

The JSNN worked with "almost 98% accuracy for most of the datasets" [32]. However, the study failed to provide any standard measures of accuracy, nor was the model compared to any other ANN models.

Goves, North, Johnston, *et al.* [57] took a different approach to improving the ANN. The majority of research considers only one geographical site or RTSS [57], whereas Goves, North, Johnston, *et al.* developed an ANN that made predictions for a network of RTSSs. This was done by improving the preprocessing of the input data. Training data of 102 traffic detectors spread over 20km of roads on the M60, M62 and M602 near Manchester, the UK with a time horizon of 15 minutes was used. The data was compressed in three ways. Firstly, the model only considered the density parameter ($\frac{volume}{speed}$), thus, making the speed and volume parameters redundant. Secondly, the data across the three lanes of a road was aggregated into one average value. Lastly, the dimensions of the data were then reduced through the use of an auto-encoder. Once the data had been compressed it was standardised (through zero-centring the data and applying a Gaussian transformation) and put through a BPNN with both one, two, and three layers. The output (predictions) was then decompressed using the auto-encoder and compared for each BPNN (one, two, and three layers). The results showed that there is a trade-off between accuracy and reducing the dimensions of the data. Nevertheless, the predictions obtained were within two vehicles/km/lane 90% of the time. However, the study only predicts vehicle density. This is a helpful indicator of the road state but traffic planners would benefit from also knowing the speed and volume. It will not be known if the density is made up of a few cars moving very slowly or lots of cars moving very fast. Also, aggregating the lanes presents a problem. Although an issue can be identified, it will not be known if it is spread across all lanes or isolated to one. Further investigations into any issues that arise would be needed.

2.2.3 Process-Based Traffic Prediction Models

The models we describe in Section 2.2.1 and 2.2.2 are all data-driven models. They are reliant on current and/or historical data to make predictions based on the analysis of previous trends and categorisations. Another class of models are process-based models (also known as model-driven models). Process-based models build a computational representation of the road network and are used to simulate road traffic systems. Unlike data-driven models, they take into account the road layout and traffic control measures such as ramp metering (traffic lights), traffic rerouting and lane closures. Therefore, process-based models can adapt to possible road layout changes and are not limited by the number of RTSSs, although, some models do use historical traffic data for calibration (introduce local bias) and bring predictions to a tolerable error level [5].

There are three main types of process-based road traffic prediction models; macroscopic, mesoscopic, and microscopic [88] (see Figure 2.9). Macroscopic models can model large areas and traffic flow. However, they fail to do the detailed traffic planning needed when issues (such as lane blockages) arise. In contrast, microscopic models are much smaller in scale and model individual drivers' behaviour and decision making. Thus, they can do detailed traffic planning but are unable to model large-scale route choices. Therefore, the final process-based model is the mesoscopic model. It is a hybrid of a microscopic and a macroscopic model. It attempts to encompass both the large-scale macrosimulations and the small-scale microsimulations [88].

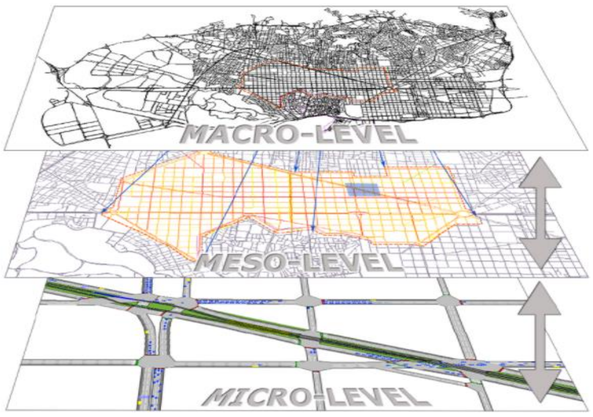

FIGURE 2.9: A depiction of the three types of process-based models; 1) Macro, 2) Meso, and 3) Micro [88]

2.3 Summary

Based on the literature review a summary table of the key characteristics, recent studies, advantages and disadvantages were produced. This can be found in Table 5.1. All models have their strengths and weaknesses, therefore, despite extensive research, several limitations and methodological gaps have been identified.

Most existing models are inaccurate. This is due to focusing too closely on one single input parameter or geographical location. For example, when considering weather as an input parameter this is done in only a single geographical location and with a single timestep. Thus, most existing research predominantly focuses on single timestep, single geographical location, and single data source or input for ease and speed of computation. To address these methodological gaps more work needs to be done exploring multiple input parameters and/or data sources as well as geographical locations.

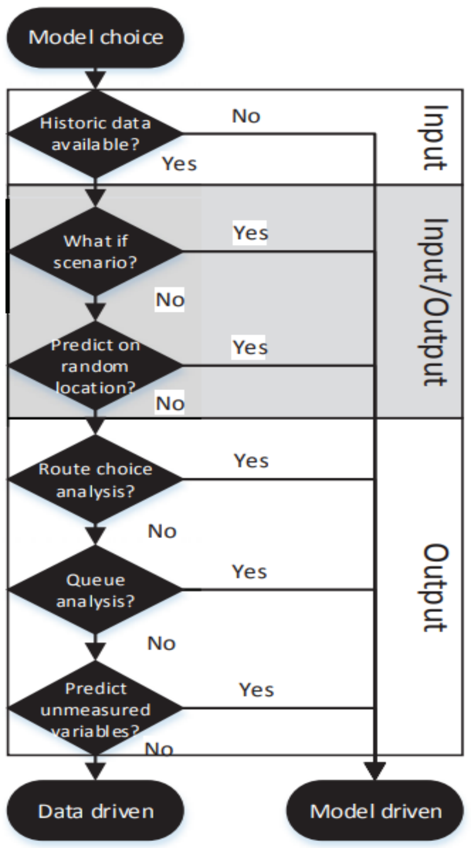

FIGURE 2.10: Questions that should be considered when choosing an approach for road traffic prediction [5]

Furthermore, there is little work done on complex road structures such as road networks and/or complex road structures such as urbanised arterial roads using abnormal traffic data. Most research is done on a simple single stretch of road or, as stated above, a single geographical location using precleaned idealistic data. This may be due to predicting road traffic congestion on such roads using such data is more complex than on other more simplistic road layouts. Their interconnected nature and composition of vehicles can vary from geographical area to time of day. Therefore, the ratio of cars to trucks can fluctuate with both vehicle classes containing their latent seasonal patterns. Research into more complex road structures using different vehicle classes as a dynamic input parameter could improve prediction accuracy. However, research on such roads and using truck flow is limited (see Figure 2.1). Therefore, this research will focus on the development of a single-step, multi-source, short-term, traffic prediction model for complex road structures.

Additionally, one aim of the literature review was to determine which model would be most suited for the prediction of road traffic flow. When choosing a short-term prediction model the motivation and context of the problem must be considered see Figure 2.10 for points that should be considered). To this end, it highlighted some critical points. Firstly, process-driven models are, in general, not suitable for short-term traffic prediction, therefore, they will not be researched further. For statistical and machine learning models the literature review highlighted that, in general, parametric models (such as ARIMA and KF) perform faster, require less data, and can model incidents previously unseen (not in historical data). However, parametric models make assumptions about the distribution of the data and are more commonly used for discrete data, not continuous. Therefore, they are not suitable for heterogeneous road traffic flow. Non-parametric models (such as ANN and KNN) on the other hand, have their structure and parameters derived from historical data. Consequently, they appear less robust than their counterparts and more data is required. However, they make no assumptions about the data distribution and complex latent non-linear patterns contained within traffic data can be modelled efficiently. Therefore, they are suitable for the prediction of road traffic flow [89] [89], thus, non-linear models should also be considered.

Past research does not agree which machine learning model is better for road traffic prediction. Some researchers state that there is no substantial theoretical reason or interest to investigate high-level non-linear mapping approaches such as ANN. This is due to the theoretical foundation of models such as the SARIMA model being adequate for prediction. Nevertheless, some researchers have argued that the use of statistical models is flawed due to their tendency to converge to the average of the historical data. Thus, the heuristic method employed by ANN is better. Therefore, due to the lack of consensus on what model is superior a selection of suitable models will be tested and compared to decide which model will be researched further.

Model	Description	Key Papers	Advantages	Disadvantages
ARIMA/SARIMA	A stochastic statistical parmetric model that assumes the input data is homoscedastic. Able to determine linear and non-linear relationships and seasonal variations.	(32), (33),(34), (35), (36), (37), (43), and (44)	Simplistic model and therefore computationally efficent. Is able to determine seasonal factors.	Assumes data is homoscedastic, thus not handle volitle data. Outliers can also skew future predictions. Reliant on historical data. The model needs to be 'fitted' to the road network and thus is not transferable.
ARCH/GARCH	A stochastic statistical non-parametric model that assumes the tha input data is hetroscedastic. Able to determine linear and non-linear relationships.	(41) (42)	Simplistic model and therefore computationally efficent. It assumes the data is hetroscedastic and thus, is robust and can handle volitile data.	Limited research on the model in traffic congestion prediction. The model must be fitted to the road network and thus, is not transferable.
GFS	A non-linear, non-function model which is able to dynamically update parameters.	(2)	Able to handle volitile traffic inout data. No fitting to the network required.	Limited research on the model in traffic congestion prediction. Due to the small 'prediction window' it requires accuate and complete traffic input data.
KNN	A non-linear, non-parametric lazy learning model that does not attempt understand the relationship between the parameters.	(11), (13), (22), (29), (47) and (48)	Simplistic model that requires no fitting to the road network so can be transferred. Robust, thus can handle volitile data.	Requires large historical dataset to search for neighbours.
KF	A non-linear or linear parametric recursive data processing model that continously updates its parameters. It assumes that the data is Gaussian distributed.	(14), (49), and (51)	Robust, thus, can handle volitile data. Only requires the previous data point for calauctions. The model allows for multivariable input.	It assumes the data is Gaussian distributed.
SVR	A non-parametric supervised learning model that maps input data into a dimensional feature space.	(55) and (16)	Able to cope with and predict abnormal traffic conditions. Simplistic model.	Large historical dataset needed to train the model.
NN	A non-parmetric non-linear flexible multivariate model that's acts as a 'blackbox' to learn complex non-linear relationships between input and ouput data.	(6) and (17)	Robust, thus, able to cope with and predict abnormal traffic conditions. High level of prediction accuracy. Is a multivariate model.	The blackbox does show the relationship between the input and the output data. It requires extensive training.

TABLE 2.3: A comparison of different statistical and machine learning models for road traffic prediction

Chapter 3

A Machine Learning Based Approach for the Prediction of Road Traffic Flow

This chapter is a modified version of 'A Machine Learning Based Approach for the Prediction of Road Traffic Flow' published in the 16th International Conference on Smart City, 2018, IEEE, and has been reproduced here with the permission of the copyright holder.

Chapter Summary

Based on the Literature Review in Section 2.3, in this chapter we have applied three different statistical/machine learning models, detailed in Section 3.2, to a real dataset for the prediction of road traffic flow on an urbanised arterial road.

Section 3.3.1 presents the comparative analysis of each model, examining their prediction accuracy and time horizon sensitivity. In Section 3.3.2 input feature settings (various classes of vehicles such as motorcycles, cars, vans, rigid goods lorries, articulated heavy goods vehicles (HGVs), and buses) were investigated to determine how heterogeneous traffic flow can affect the prediction results.

3.1 Introduction

An urbanised arterial road is a major high capacity and large traffic volume road located in a built-up area. Urbanised arterial roads have limited entry and exit points and are used as thoroughfares for travelling substantial distances between major geographical locations, such as motorways or city centres. Therefore, they are often used for commuting and transportation of goods. The importance of keeping these roads free of congestion is imperative, as severe congestion can result in traffic delays extending into various other traffic networks. Due to a high volume of heterogeneous traffic, arterial roads tend to be consistently busy and contain different seasonality patterns across different vehicle classes when compared with motorways or city centres. Additionally, due to operating at close to capacity levels, their traffic flow tends to be more volatile. Therefore, research into traffic prediction models for this class of road is vital. One key component of arterial roads which could improve the prediction is the latent patterns embedded within the different vehicle classes. However, there is a lack of research on arterial roads, particularly using vehicle classes as a variable. Machine learning models are a popular option for traffic prediction [44], which are capable of learning complex and latent patterns embedded in volatile traffic data containing both normal and abnormal traffic congestion.

The contributions of this chapter include:

1. we have examined and compared the traffic flow prediction accuracy and time horizon sensitivity of existing machine learning models for urbanised arterial roads, based on a real dataset; and

2. we have investigated different input parameter settings (different classes of vehicles) to understand how heterogeneous traffic flow affects the machine learning models' prediction accuracy for urbanised arterial road

3.2 Methodology

In this section, we are mainly focusing on two goals: 1) comparing the performance of existing machine learning models to determine which is superior to use for urbanised arterial roads between Manchester and Liverpool, and 2) determining if the use of vehicle classes as separate input parameters can improve the prediction of road traffic flow on urbanised arterial roads with heterogeneous traffic flow.

To achieve the first goal, two experiments were carried out. Firstly, we have applied three machine learning models to a real dataset and compared the accuracy of their predictions. Furthermore, the second experiment compared the models' sensitivity to time horizon changes. As previously stated in Section 2.1.1, there is no standard time horizon for short-term road traffic prediction. Therefore, in the second experiment, each model was run with a time horizon of five, ten, fifteen, and twenty minutes. Their results were analysed and compared to determine how the magnitude of the time horizon affected the prediction accuracy.

For the second goal, to the best of our knowledge, no research has investigated the effect of all vehicle classes on the accuracy of road traffic prediction. Therefore, two experiments were carried out. Firstly, an experiment was performed to compare the prediction of traffic flow using only the total vehicles and the individual vehicle class totals. In addition, a second experiment was conducted comparing the prediction accuracy using each vehicle class to determine which class or classes were most influential for prediction. The details of all experiments conducted using Matlab are detailed below.

The three machine learning models chosen for experimentation were KNN [22], SVR [23], and ANN [24]. These models make no assumptions about the distribution of the data (non-parametric), are suitable for heterogeneous data, and can cope with volatile data. Therefore, they are suitable for the prediction of road traffic flow [89].

3.2.1 KNN Model

The prediction function used was the inverse distance weighting (IDW) function (Equation 3.1). This function gives priority to closer neighbours than those further away, thus, making it more reliable for predicting extreme traffic conditions (if these conditions have been seen before).

The IDW can be calculated as

$$\hat{x} = \frac{\sum_{i=1}^{k} x_{i,t+h} w_i}{\sum_{i=1}^{k} w_i} \tag{3.1}$$

Where $\hat{x}$ is the predicted value, x_i is the neighbour from the training set, t is the time parameter, h is the time horizon, and w is defined as $\frac{1}{d_i}$, x_i is the neighbour, and d is the distance function.

During the experiments, the distance function used was the Euclidean distance function (Equation 2.20). This was due to all input parameters being of the same type (number of vehicles). The values of k used were between 30 and 90 in increments of 5 and determined through brute force by a grid search to find the optimal value. The different architecture structures were trained using two months of data and tested using one month of data (hold-out cross-validation), as described in Section 3.2.4. Each variation of the model architecture was trained and the empirical error, based on the performance metric described in Section 3.2.5, was taken. The architecture that produced the lowest empirical error was used for comparison.

3.2.2 SVR Model

In the SVR model the dot product in Equation 2.30 was replaced with the RBF kernel function (Equation 2.35), the most commonly used function for road traffic predictions due to its powerful non-linear learning ability [90] [84]. It has strong learning ability but weak generalisation, allowing the model to better handle volatile traffic data, as extreme historical variances will not negatively affect future predictions. Additionally, Cherkassky et al. [91] investigated the selection of hyperparameters for SVR models. They concluded that the parameter selection of c, which balances the model's complexity and estimation error, and ϵ, which is used to fit the training data, can be derived directly from the training data. Therefore, the values of c and ϵ used in the experiments can be found using Equations 3.2 and 3.3 [91].

$$c = max(|\hat{y} + 3\sigma_y|, |\hat{y} - 3\sigma_y|) \tag{3.2}$$

$$\epsilon \in [0, 0.1] \tag{3.3}$$

The value of ϵ was determined through brute force by a grid search to find the optimal value. The different architecture structures were trained using two months of data and tested using one month of data (hold-out cross-validation), as described in Section 3.2.4. Each variation of the model architecture was trained and the empirical error, based on the performance metric described in Section 3.2.5, was taken. The architecture that produced the lowest empirical error was used for comparison.

3.2.3 ANN Model

The ANN model [24] used in the experiment was an open loop time series non-linear autoregressive network with exogenous inputs (NARX). NARX is a popular ANN model used in time series prediction. It can predict traffic flows efficiently as it bases its predictions on historical data, feedback (recurrent), and exogenous input (current and historical data). Therefore, it can 'remember' the recent past and handle volatile data [32]. The transfer function used was a Sigmoid Transfer Function (Equation 3.4). This is a standard transfer function used for time series data commonly used for road traffic flow prediction.

$$S(x) = \frac{e^x}{e^x + 1} \tag{3.4}$$

All weights and biases were randomly initialised and a dropout rate of 50% was used based on research by Zhao, Chen, Wu, *et al.* [33] and Srivastava, Hinton, Krizhevsky, *et al.* [92]. All other hyperparameters, such as the number of layers and nodes, were determined through brute force by a grid search to find the optimal design. The different architecture structures were trained using two months of data and tested using one month of data (hold-out cross-validation), as described in Section 3.2.4. Furthermore, the Adamax optimiser [93] was used to optimise the model's learning rate based on the data's characteristics (a separate step size, known as the learning rate, for each parameter). Each variation of the model architecture was trained and tested 10 times and an average empirical error, based on the performance metric described in Section 3.2.5, was taken. The architecture that produced the lowest average empirical error was used for comparison.

3.2.4 Dataset Description

The data used for the experiments was a real-life dataset of an urbanised arterial road between Manchester and Liverpool, UK. The dataset contained three months of traffic flow data collected between 1st January to 31st March 2016, broken down into vehicle classes, as shown in Table 3.1, and contains a time horizon of five minutes (26,195 data points). Therefore, the input parameter used in the experiments were the different vehicle classes. This data was used due to being a typical urbanised arterial road. It has a heterogeneous traffic flow, containing high proportion of trucks and HGVs typically found on these road types. Furthermore, the data contains both normal and abnormal traffic data.

For experimental purposes hold-out cross-validation was used to split the data. Two months of data (18,337 data points) were used to train the models and the remaining one month (7,858 data points) was used for testing purposes.

Class No.	Vehicle Type
1	Motorcycles
2	Car or Van
3	Car or Van with Trailer
4	Rigid Goods
5	Articulated HGV
6	Bus or Coach

TABLE 3.1: The different categories of vehicle types

3.2.5 Performance Metrics

To compare each model the standard performance metric RMSE, as shown in Equation 2.3, was used to measure the average deviation between the predicted value and the actual value.

3.3 Experimental Evaluation

3.3.1 Comparison of Existing Machine Learning Models

To achieve goal one the following two experiments were performed:

- Experiment One: The machine learning models were compared on their empirical prediction accuracy, using the same real dataset, to determine which model performed the best.

- Experiment Two: The machine learning models were compared on their empirical accuracy, using various time horizons (five to twenty minutes in five-minute intervals) and the same real dataset, to determine the models' time horizon sensitivity.

Experiment One

Table 3.2 shows the set up parameters used during each run of Experiment One. It should be noted that the value of k in the KNN model was 65, the value that produced the lowest empirical root-squared mean error (RSME), in all experiments.

TABLE 3.2: The parameter set up for experiment one

Parameter	**Details**
Time Horizon	5 Minutes
Training Data Size	2 Months
Test Data Size	1 Month
Input Parameters	All Vehicle Class Totals
Parameter Changed	Model Only

Experiment One Results

In Figures 3.1, 3.3, and 3.5, the red lines represent the predicted and blue lines the actual traffic flow values. The y axis is total number of vehicles, which is plotted against the x axis, time. Experiment One's results (Table 3.3) show that the most successful traffic prediction model for the urbanised arterial road between Manchester and Liverpool was the ANN model, with a RMSE of 16.56% compared to the KNN and SVR which had a RMSE of 19.35% and 19.79% respectively.

TABLE 3.3: The RMSE of the KNN, SVR, and ANN model

Model	**RMSE (%)**
KNN	19.35
SVR	19.79
ANN	16.56

Experiment Two

Table 3.4 shows the set up parameters used during each run of Experiment Two.

Experiment Two Results

Table 3.5 and Fig 3.7 presents the RMSE (in percentage form) for all models and time horizons. The results show that the most successful time horizon was the shortest, five minutes. Although not explicitly stated in publications, this may be the rationale behind why five minutes is the most commonly used. ANN, again, was the

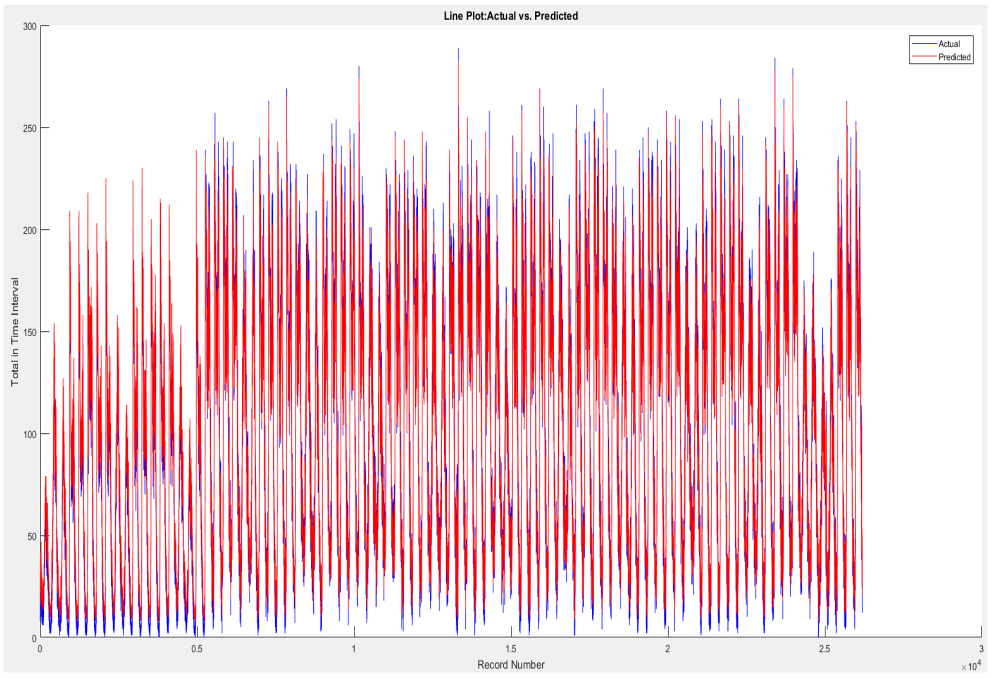

FIGURE 3.1: The KNN model's actual and predicted road traffic flow plotted against time

TABLE 3.4: Parameter set up for experiment two

Parameter	**Details**
Time Horizon	5 to 20 Minutes
Training Data Size	2 Months
Test Data Size	1 Month
Input Parameters	All Vehicle Class Totals
Parameter Changed	Model & Time Horizon

most successful prediction, however, it was also the most sensitive to time-horizon changes, with the RMSE increasing by 4.64%. KNN and SVR in comparison appear to fluctuate with the general trend, based on the limited data, shows that the prediction accuracy progressively declines.

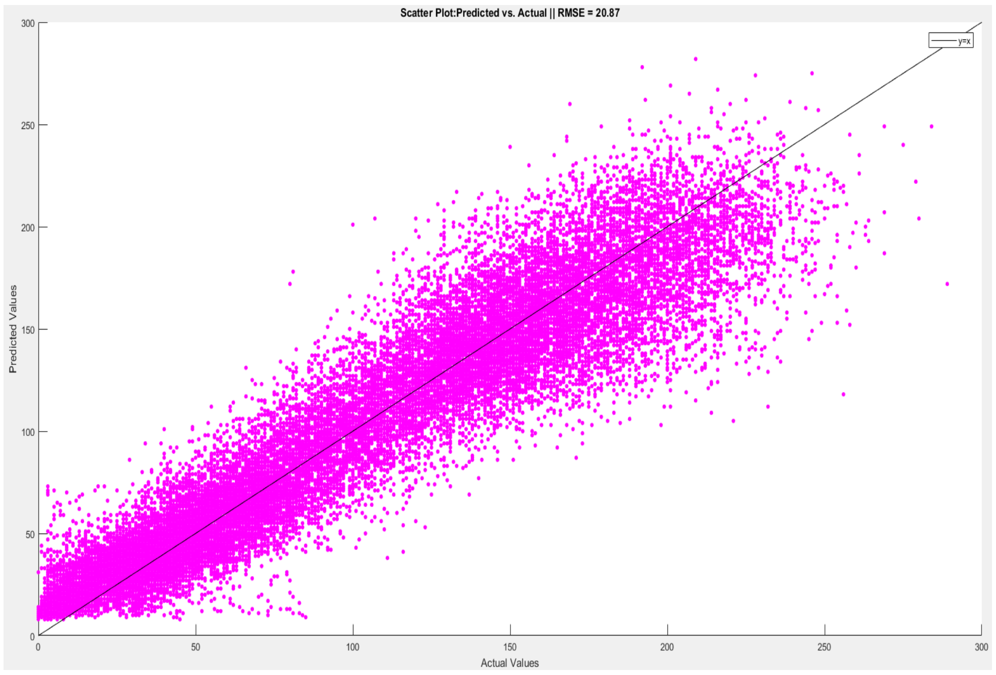

FIGURE 3.2: The KNN model's actual vs predicted road traffic flow

TABLE 3.5: The time horizon sensitivity of the KNN, SVR, and ANN models measured by their RMSE

Model	**5 Minutes**	**10 Minutes**	**15 Minutes**	**20 Minutes**
KNN	19.35	18.68	19.60	19.10
SVR	19.79	20.37	21.50	21.42
ANN	16.56	17.25	18.54	21.20

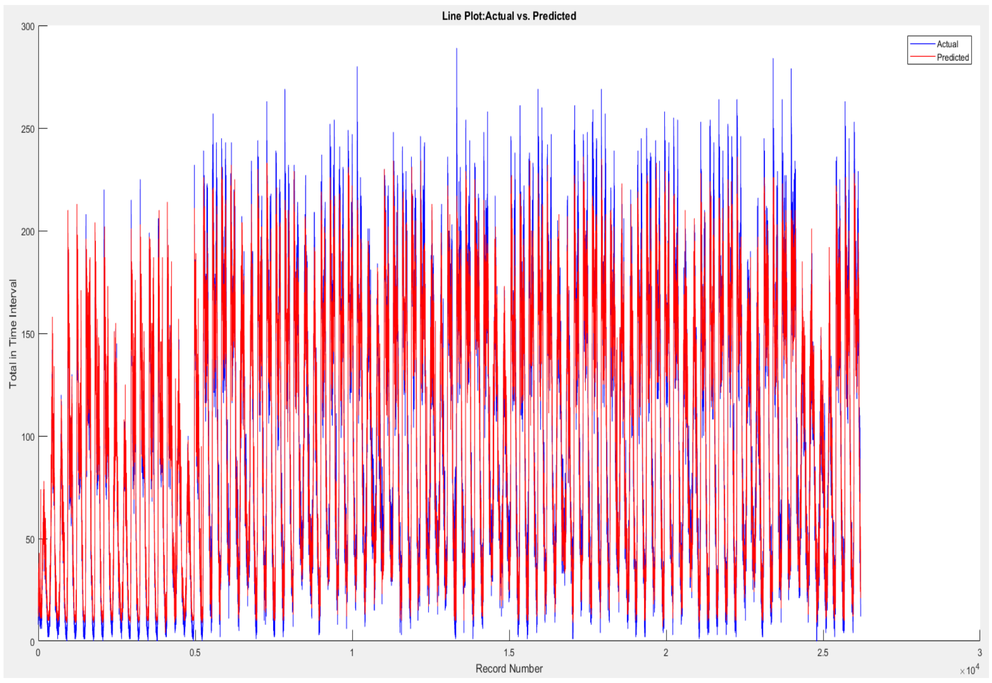

FIGURE 3.3: The SVR model's actual and predicted traffic flow plotted against time

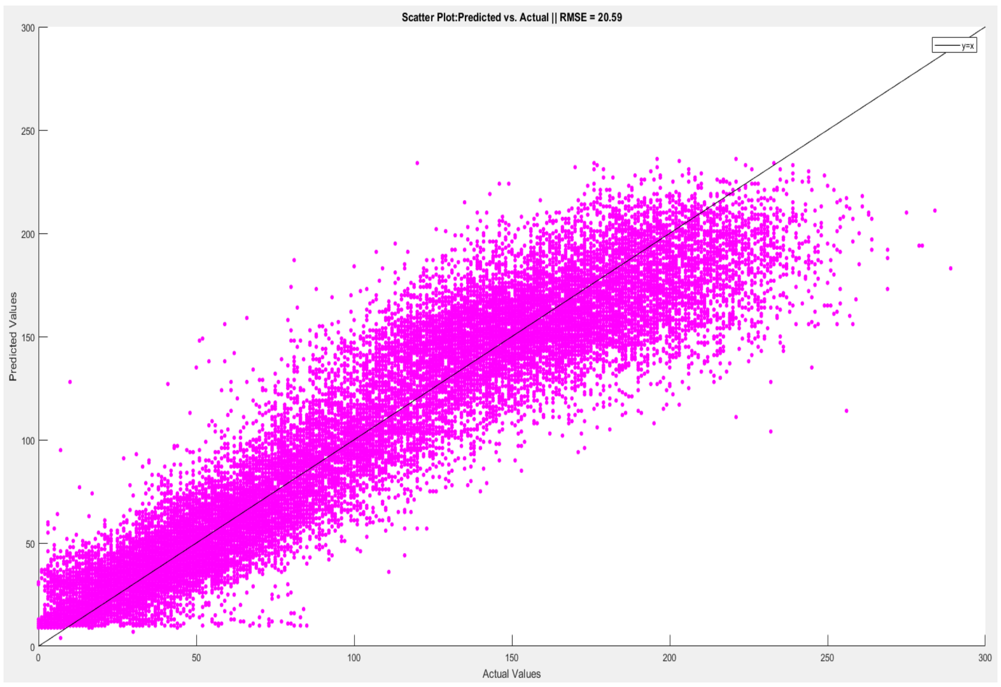

FIGURE 3.4: The SVR model's actual vs predicted road traffic flow

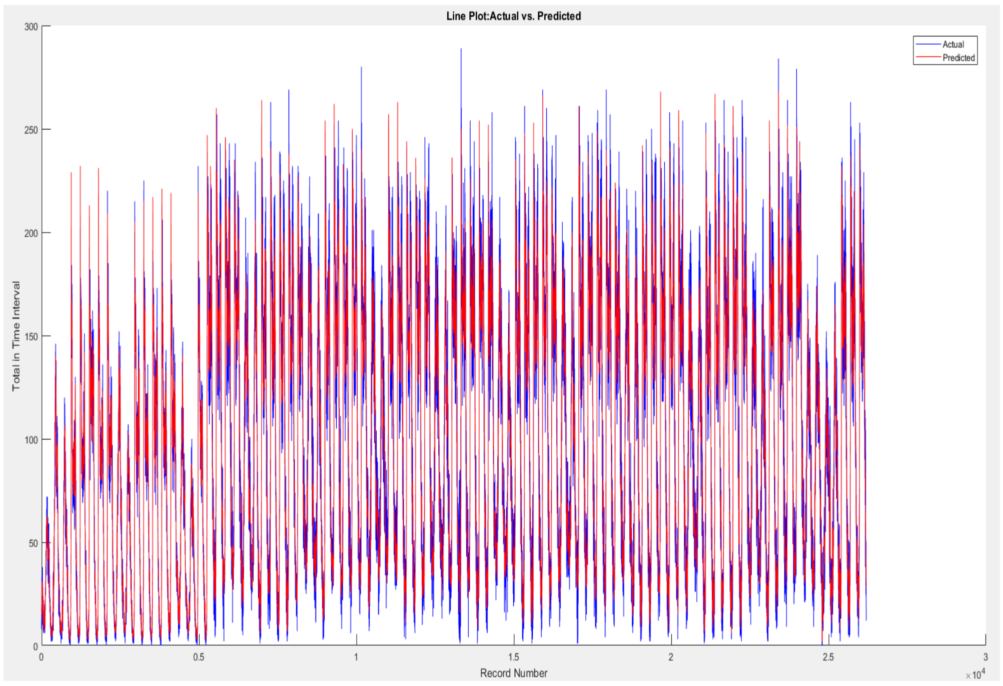

FIGURE 3.5: ANN model's actual and predicted traffic flow plotted against time

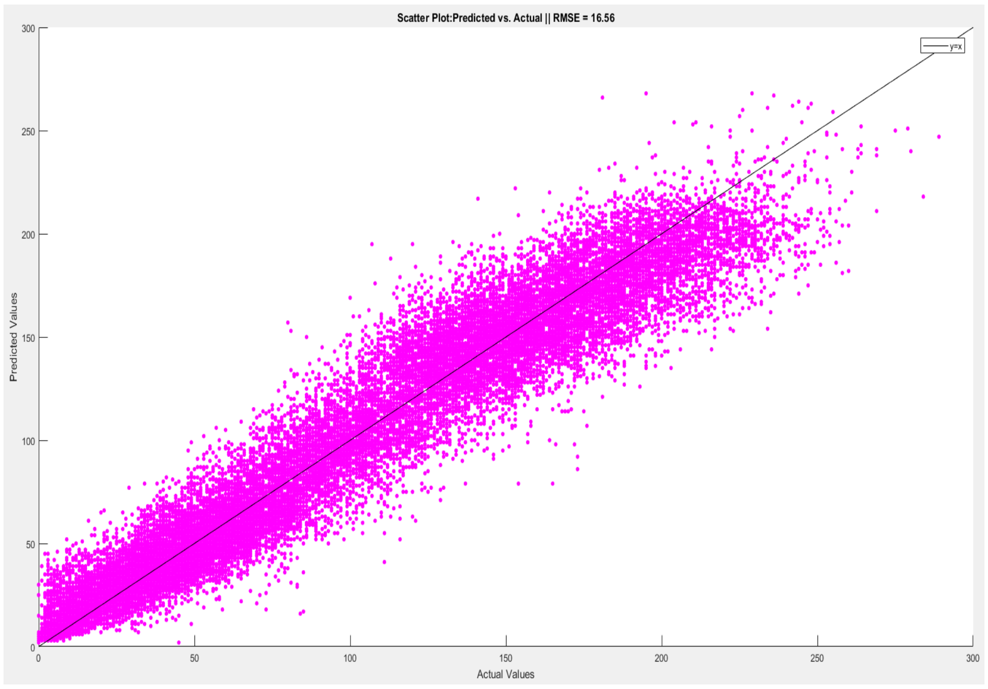

FIGURE 3.6: The ANN model's actual vs predicted road traffic flow

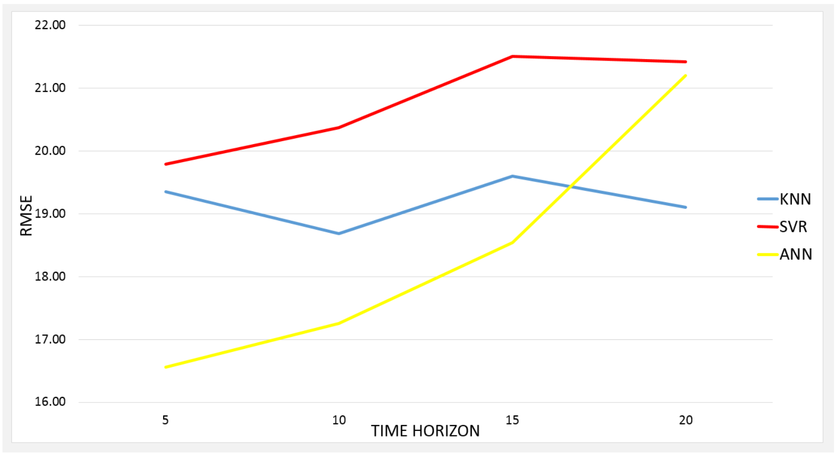

FIGURE 3.7: The time horizon sensitivity of the KNN, SVR, and ANN model shown by their RSME vs the time horizon

3.3.2 The Effect of Heterogeneous Traffic Flow on Prediction Accuracy

To achieve goal two, determine if the use of vehicle classes as separated input parameters can improve the prediction of road traffic flow on urbanised arterial roads between Manchester and Liverpool, using real-life data, the following two experiments were performed:

- Experiment Three: The machine learning models were compared using the same real dataset while changing the input parameters from total traffic flow to all individual vehicle classes' total traffic flow. This was done to determine if the latent patterns of each vehicle class would affect the prediction accuracy.
- Experiment Four: The machine learning models were compared using the same real dataset while using one vehicle class total as an input parameter. This was done to determine which vehicle classes were most influential in predicting traffic flow.

Experiment Three

Table 3.6 shows the set up parameters used during each run of experiment three. Each model was tested and compared using a different input parameters. The input parameters used were the total vehicle traffic flow (a singular input parameter) or the individual vehicles class totals (six input parameters).

TABLE 3.6: The parameter set up for experiment three

Parameter	Details
Time Horizon	5 Minutes
Training Data Size	2 Months
Test Data Size	1 Month
Input Parameters	All Vehicle Class and Total Traffic Flow
Parameter Changed	Model & Vehicle Classes Total

Experiment Three Results

Table 3.7 shows that, for all models, using the individual vehicle class totals as an input parameter improves traffic flow prediction when compared to predictions based only on total traffic flow. Therefore, taking into consideration the latent patterns for each vehicle class does improve prediction accuracy.

TABLE 3.7: The comparison of the KNN, SVR, and ANN models' RMSE (%) based on the input parameters of total vehicle flow or total of all classes of vehicles

Model	Input of All Vehicle Classes	Input of Total No. of Vehicles
KNN	19.35	20.01
SVR	19.79	19.89
ANN	16.56	16.95

Experiment Four

Table 3.8 shows the set up parameters used during each run of experiment four. Each model was tested and compared using a different singular input parameter. The input parameters used were the total of all classes, Class One, Class Two, Class Three, Class Four, Class Five, or Class Six of vehicle traffic flow. It should be noted that the value of k in the KNN model was 65, the value that produced the lowest empirical root-squared mean error (RSME).

TABLE 3.8: The parameter set up for experiment four

Parameter	Detail
Time Horizon	5 Minutes
Training Data Size	2 Months
Test Data Size	1 Month
Input Parameters	One Vehicle Class
Variable Changed	Model & Individual Vehicle Class

Experiment Four Results

Based on Experiment Four's results, Class Two (car or van) is the most influential vehicle class for road traffic prediction, followed by Class Four (rigid goods) for all models. It is also worth noting that the results are better when only considering Class Two vehicle total, as oppose to all vehicle class totals.

TABLE 3.9: The RMSE of the KNN, SVR, and ANN model using different classes of vehicles as input parameters for the prediction of road traffic flow

Model	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
KNN	19.96	18.60	20.01	19.92	20.01	20.01
SVR	19.94	18.53	19.99	19.90	20.01	20.00
ANN	16.84	16.40	16.92	16.83	16.93	16.85

3.4 Summary

In this chapter, we have conducted a comparative analysis of three machine learning algorithms (KNN, SVR, and ANN). The three machine learning algorithms were applied to a real dataset to determine: 1) which machine learning model produced the most accurate prediction and how time horizon changes affect their accuracy (time horizon sensitivity); and 2) the impact of vehicle classes as input parameters on prediction accuracy.
The experimental results presented show that the most accurate machine learning model was the ANN. It consistently produced the most accurate prediction. However, the ANN model was the most sensitive to time horizon changes, as seen in Table 3.5, but still produced the most accurate prediction in all time horizons tested. Therefore, it can be concluded that ANN models are suitable for short-term traffic flow prediction on urbanised arterial roads. ANN should be explored further for the prediction of road traffic flow.

The experimental results also show that using the individual vehicle class totals, as opposed to the total number of vehicles, improves all three machine learning models' predictions (as seen in Table 3.7). This is likely due to the models being able to ascertain latent patterns within the different vehicle class data to improve prediction accuracy. The composition of vehicles on urbanised arterial roads varies from geographical area to time of day. The ratio of cars to trucks can fluctuate with both classes having their latent seasonal patterns. Therefore, considering each class as a separate input parameter has improved the prediction accuracy. In addition, the most influential vehicle classes were Class Two (car or van) and Class Four (Rigid Goods). This may be due to these particular classes being the main classes in terms of volume in the data. However, it is worth noting that prediction accuracy improved when considering only Class Two as an input parameter compared with using all vehicle class totals. This suggests that the latent seasonal pattern within the main vehicle class type (Class Two in this research) is key to the prediction of road traffic flow.

Based on the experimental results the prediction of road traffic flow should focus on ANNs. Different ANN architectures should be explored and their suitability for road traffic flow prediction assessed, especially ANNs with deep architecture. Deeper architectures could allow an ANN to obtain a more complex, non-linear function and therefore, improve prediction accuracy. Deep neural networks, such as deep recurrent neural networks, are now being explored for road traffic flow prediction. However, what deep architecture is the most appropriate remains unanswered. Previous research into deep recurrent neural networks fails to compare them to other deep models; instead, comparisons are made with simple shallow models. To compound this issue standard performance metrics, such as RMSE, assess a model's success solely on its accuracy. No consideration is given to computational cost. A model must be assessed on both its accuracy and speed. Another issue is the optimisation of a neural network's architecture can be difficult. There is no standard or analytical method to determine their correct structure. This often leads to sub-optimal architectures being used. Therefore, deep neural networks (DNNs) should also be assessed on how sensitive the model is to architectural changes.

Chapter 4

Evaluating the Performance of Deep Recurrent Neural Networks

This chapter is a modified version of 'Prediction of Road Traffic Flow Based on Deep Recurrent Neural Networks' published in the 5th IEEE Smart World Congress, Leicester, UK, 2019, IEEE, and has been reproduced here with the permission of the copyright holder.

Chapter Summary

Based on the experimental results in the previous chapter (see Section 3.3), in this chapter different deep ANN architectures were explored for their suitability for the prediction of road traffic flow. Three deep recurrent neural networks, described in Section 4.3.2, were examined on their accuracy, training time, and sensitivity to architectural change. The results of these experiments are presented in Section 4.3.3. Additionally, a new performance metric was developed, Standardised Accuracy and Time Score (STATS), shown in Section 4.3.3, which standardises both the accuracy and computational time into a comparable score, allowing an overall score to be awarded.

4.1 Introduction

Due to advances in computing power and algorithm development by Hinton et al. [94] the depth of neural networks is increasing, leading to superior performances. Deep neural networks (DNNs) [95] are now feasible and more efficient for large complex data [96]. They are favoured over shallow architectures due to their ability to efficiently extract complex latent patterns embedded within the data [97] owing to their long computational chain of layers [98]. Despite this, neural networks designed for road traffic prediction are predominately shallow architectures with only one hidden layer [99]. Therefore, research into exploring deep architectures to improve prediction accuracy for road traffic flow is now possible and needed.

There is a wealth of neural network architectures and hybrids. The real challenge in the DNN domain is model selection [100]. Which deep architecture is the most appropriate for noisy time series data, such as road traffic flow, remains unanswered [38]. To compound this issue more, there is a lack of research comparing deep learning models for road traffic flow prediction. Researchers often compare their models to simple shallow architectures or statistical models which can not compete with the long computational chain of layers of DNN. Another major flaw of DNN research is its disregard for the computational cost (training time). It is obvious that the deeper a neural network, the bigger the computational cost, however, this is often omitted from evaluations. Traffic flow prediction models assess success solely on prediction accuracy; predominately the RMSE or mean absolute percentage error (MAPE) is used to evaluate a model [44]. No evaluation of the training time taken to produce the prediction is made. Additionally, no trade-off between the accuracy and training time is investigated. The current standard performance metric is antiquated. Successful models should produce tolerably accurate results within a reasonable time frame set by the specific problem domain. Therefore, a new performance metric is needed for DNNs.

Furthermore, to fairly compare different DNNs each model should be fully optimised to achieve its best prediction, yet there is no standard procedure or analytical calculation to determine the optimal architectural structure. Hyperparameters are optimised using domain knowledge through experience and literature review, or heuristics through grid search style algorithms. Many model optimisations are difficult, not consistent, and at times not appropriate for the domain problem. Therefore, assessing how sensitive a model is to architectural change is important. It should be established how consistently a model produces an acceptable prediction within a tolerable time frame set by the domain problem.

The contributions of this chapter include: 1) we have examined and compared three existing deep recurrent neural networks (a basic recurrent, a long short-term memory, and a gated recurrent unit), using a novel performance metric, based on a real dataset to determine which is suitable for road traffic prediction, 2) for a fair comparison, a new performance metric, the standardised accuracy and time score (STATS) was developed to assess the overall performance of the DNNs based on their prediction accuracy and training time, and 3) we have examined the DNNs' sensitivity to architectural changes.

4.2 State of the Art Deep Learning Traffic Flow Prediction Models

In this section, we will review and assess deep recurrent architectures for their suitability for the prediction of road traffic flow.

Lv, Duan, Kang, *et al.* [99] is, to the best of our knowledge, the first publication on DNNs for road traffic prediction. Lv, Duan, Kang, *et al.* stated that shallow prediction models with limited hidden nodes do not learn an adequate representation of the relationship between the current and the future traffic flow due to the data's stochastic and complex non-linear properties. Instead, a compressed version is found. Thus, a DNN is needed; however, they do have limitations. Adding unneeded layers can negatively affect the model's prediction accuracy and increase training time.

More, Mugal, Rajgure, *et al.* [32] used traffic flow data from Dublin, Ireland, with a five-minute timestep and a Jordan neural network model (JNN) [101] to predict road traffic flow. A JNN is an adaptation of an RNN, as decribed in Section 2.2.2, that can be identified by the inclusion of a *context unit*. The context unit receives the previous timesteps prediction from the output layer (y_{t-1}) to provide *context* for the current prediction (RNNs receive context from the previous timestep's hidden state (h_{t-1})). More, Mugal, Rajgure, *et al.* concluded that the structure of an ANN determined how successful a prediction was, and reported "almost 98% accuracy for most of the datasets" [32]. However, the study failed to compare the model to any other ANNs. Other variations of RNNs may be more suitable. State-of-the-art research using RNNs for time series prediction has focused on other adaptions of RNNs due to the basic structure of RNNs and JNNs being unable to capture the long-term dependencies within time series data [102]. An RNN can not learn dependencies if the time lag is greater than 5-10 discrete timesteps apart [103]. This severely limits the range of contextual information of an RNN [104]. Furthermore, the success of an

RNN is hindered by vanishing gradient problems [105]. Therefore, it is difficult to find advantages over feed forward neural networks.

Long short-term memory (LSTM) neural networks [106] were initially developed to solve the *vanishing/exploding gradient problem* through the introduction of a *memory* called the *cell* and the use of a *constant error carousel* (CEC). An CEC denotes the recurrent connection of the *cell state* (c_t), where an error can flow through *unchanged*, as depicted in Figure 4.1 by the directed circle encompassing the word 'cell'. Therefore, in an LSTM an error is reduced when it is backpropagated through the multiple layers, however, it is not modified in the CEC as it is designed to preserve the error through time and layers. Preserving the error allows the model to continue to learn over many timesteps, therefore, linking cause and effects which are many timesteps apart.

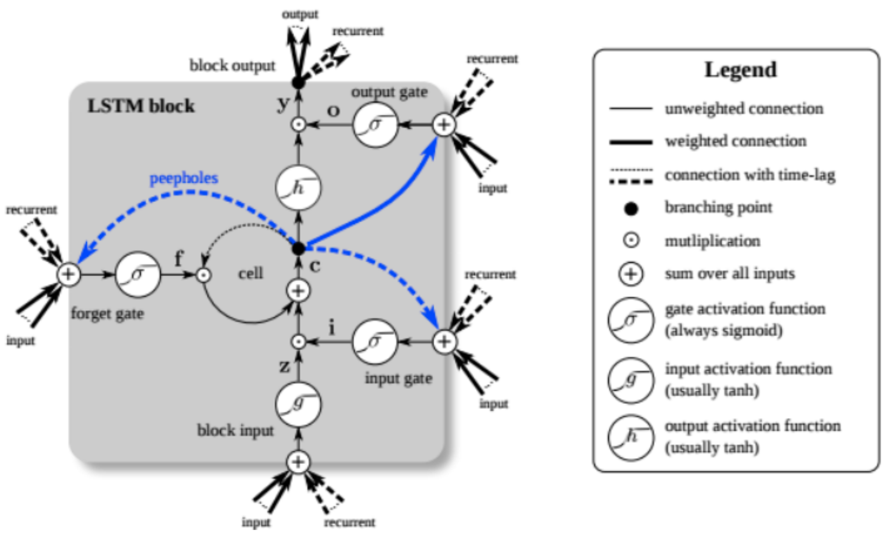

FIGURE 4.1: A detailed schematic of a long short-term memory block (also known as a cell) [107]

Zhao, Chen, Wu, *et al.* [33] used an LSTM for the prediction of road traffic flow using temporal-spatial correlations. The traffic data used was 19 days, split into five-minute timesteps, of 500 observation points on one road segment (although only three points were used for prediction). The paper pre-processed the traffic data using an *origin-destination cost* (ODC) matrix to find the correlations between the temporal and spatial data. Once the data is preprocessed it passed through a standard LSTM. The model was compared using four different LSTMs with two, three, five, and six layers to find its optimal structure. It was also compared to an ARIMA, SVM, RBF, SAE, and a standard RNN. The paper reported that preprocessing the input data improved accuracy and the LSTM was the most accurate. However, the trade-off between training time and accuracy was not considered. Is the additional computational time to preprocess the data justified by the increase in accuracy? Furthermore, it is not explicitly stated if the correlation data was also used in the comparison models.

Shi, Xu, and Li [108] used an LSTM to predict household loads. Shi, Xu, and Li stated that the main problem with DNNs is their tendency to overfit to training data and therefore, results can not be generalised. Furthermore, the input data, household loads, is noisy due to external factors. To overcome these problems Shi, Xu, and Li preprocessed the input data in a *pooling layer*. The pooling layer is designed to compensate for the small amount of temporal data by adding a new input parameter to the target household. The new parameter added is the neighbour's household load. By adding the neighbour's household load the input data has more diversity.The training data from 929 households divided into pools of ten were used, which consisted of 48 hours of house loads with a timestep of 30 minutes. The pooled training data was passed through LSTM and its predictions were compared with an ARIMA, SVR, and an RNN. It was reported that the model's accuracy, in terms of RMSE, was superior. The main issue with the paper is the problem it seeks to overcome, the overfitting of DNNs. If a model is overfitting the regularisation term needs to be adjusted, or other methods, such as dropout rates and early stopping of training, can be used. The paper has failed to mention these methods or address their inadequacies and state why they have chosen to develop a new method. The paper also fails to address the increase in training time due to the addition of a pooling layer.

In conclusion, although the internal memory of an LSTM has proven it can remember data over many time lags it is yet to be demonstrated that they can perform complex rational reasoning with this information. Many models still adjust their input data to provide more information to aid the models' prediction accuracy. Furthermore, due to their complex cell structure, as seen in Figure 4.1, the training of an LSTM has high computational costs. No paper has addressed the trade-off between the training time and accuracy.

Cho, Merrienboer, Gulcehre, *et al.* [34] put forward another adaptation of an RNN to solve the vanishing gradient problem, a gated recurrent unit (GRU) neural network. Similar to the LSTM, the GRU can be trained to retain information over many time lags through the use of gates. GRUs are still in their infancy, therefore, there is limited research regarding them, with most papers performing comparative studies. Fu, Zhang, and Li [38] conducted a comparison study of an ARIMA, LSTM, and an GRU. The study used four weeks of data from 50 RTSSs in Oakland, USA, with a training and test split of three weeks and one week respectively, and a timestep of five minutes. Fu, Zhang, and Li reported that the GRU model was, in terms of mean squared error (MSE), the most accurate for prediction of road traffic flow. However, no other architectures of DNNs were compared, and the models' training times were not considered.

Similarly, Zhang and Kabuka [109] also performed a comparison study using the same data as the paper above but also included weather data such as wind speed, precipitation, and temperature. The study compared an ARIMA, a SVM, an RNN, LSTM, and a GRU model. The study reported that, in terms of RMSE, the GRU model was the most accurate. Furthermore, the model was compared with and without the weather data and discovered that its inclusion improved the model's prediction by 25%. Again, the paper only compares the accuracy of the neural network architectures. No training time was considered. Furthermore, the models' sensitivity to architectural change was not investigated. The GRU model may have produced the most accurate prediction, however, it is not known how volatile its predictions are.

In conclusion, RNNs are designed for time series prediction and have provided good prediction results. However, there is still a lack of research comparing deep learning models. Many comparisons are still made with simple shallow architectures which can not compete with DNNs' long chain of computational layers [108]. Furthermore, common adaptations of RNNs tend to focus on how best to preprocess and include different input parameters to improve the model's accuracy. No consideration is given to the trade-off between the accuracy and training time. Therefore, model selection remains the real challenge [100] for noisy time series data such as road traffic flow [38].

4.3 Methodology

The main aims of this chapter are to compare three existing deep RNNs to determine which is most suitable for road traffic prediction, and to examine their sensitivity to architectural changes using a performance metric that can assess their overall performance. To achieve these aims we have focused on three research questions: 1) what is the most suitable deep RNN for road traffic flow prediction based on its accuracy and training time, 2) what is the most appropriate performance metric for deep neural networks, and 3) which deep RNN is the least sensitive to architectural change (the model's range of predictions over various architectural structures)?

To answer all three research questions 36 experiments were conducted utilising three deep recurrent learning models (12 per model) using different architectural setups and a real dataset. The prediction accuracy and training time were compared and analysed.

4.3.1 Data Description

The deep learning models were applied to a real-life dataset from an arterial road between Manchester and Liverpool, UK. The dataset consisted of three months of data collected between 1st January to 31st March 2016, with a time horizon of five minutes (26,195 data points) and input parameters of different vehicle classes as shown in Table 3.1. Two months of data (18,337 data points) were used for training and validating the models, and one month of data (7,858 data points) was used for testing the models.

4.3.2 The Model Architectures

The deep recurrent learning models chosen for comparison and used during experimentation were:

1. An RNN model
2. An LSTM model
3. An GRU model

All weights and biases were randomly initialised and a dropout rate of 50% was used based on research by Zhao, Chen, Wu, *et al.* [33] and Srivastava, Hinton, Krizhevsky, *et al.* [92]. All other hyperparameters, such as the number of layers and nodes, were determined through brute force by a grid search to find the optimal design. Specifically, all models were set up using a range of 12 different architecture structures consisting of two, three, five, and six layers (excluding any input and output layer) [33] with 12, 16, and 20 nodes in each layer. The different architectural structures and their variations were trained using two months of data and tested using one month of data (hold-out cross-validation), as described in Section 4.3.1.Furthermore, the Adamax optimiser [93] was used to optimise the model's learning rate based on the data's characteristics (a separate step size, known as the learning rate, for each parameter). Each variation of the models' architectures was trained and tested 10 times and an average empirical error, based on the performance metric described in Section 4.3.3, was taken. The architecture that produced the lowest average empirical error was used for comparison. Below each model is explained in detail.

RNN Model

A RNN is designed to detect sequences in temporal data by the assuming the input and output pairs are dependent on each other, meaning the model takes into account the previous prediction when calculating the current prediction. Therefore, the order of the sequence is taken into consideration; the temporal element of the input data adds a new dimension to the prediction function being learned. The model achieves this through the use of *loops*. RNNs contain a *hidden state node* (s) that feeds back the previous timestep ($t-1$) value. This allows the model to find and use correlations between the temporal data that may be many timesteps apart, as seen in Figure 4.1, and is often referred to as an *internal memory*. Therefore, the current hidden state is a function of current input data (x_t) and the previous timestep's (h_{t-1}) hidden state, as seen in Equation 4.1, where h_t is the hidden state, t relate to the timestep, σ is the activation function, x_t is the input data at time t, and u is the weight matrix.

$$h_t = \sigma_h(u_h x_t + w h_{t-1} + b_h) \tag{4.1}$$

Therefore, the prediction (y_t) is calculated as shown in Equation 4.2.

$$y_t = \sigma_y(w_y h_t + b_y) \tag{4.2}$$

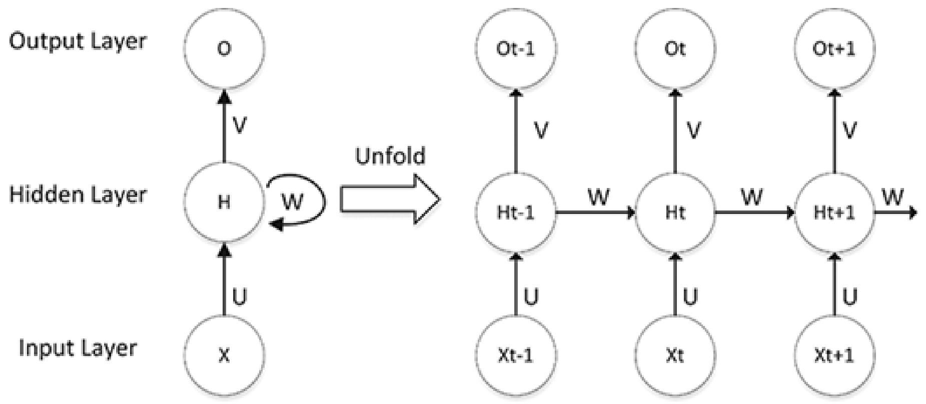

FIGURE 4.2: A basic RNN with its hidden state unfolded [110].

LSTM Model

A standard LSTM NN *cell* contains three *gates*, a *forget gate* (f), an *input gate* (i), and an *output gate* (o). It contains a hidden state (h) and a *memory state* (C), as seen in Figure 4.1.

The memory state is the main concept behind the LSTM NN; it transfers the data through the cell via a sequence of gates. The gates decide what information in the cell state should be added or removed. Each gate is a single-layered neural network with a sigmoid activation function (σ), as shown in Equation 3.4. The sigmoid function outputs a value between zero and one (see Equation 4.3) to determine how much or how little should be added or deleted from the cell state.

$$\sigma(x) = \frac{e^x}{e^x + 1} \tag{4.3}$$

The cell state is first computed by calculating the block input (z) based on the previous hidden state (h_{t-1}) and the current input data (x_t), as shown in Equation 4.4, where w is a weight and b is a bias. An activation function, usually tanh (as shown in Equation 4.5), is used to regulate the network by squashing the input values between -1 to 1.

$$z = \tanh(w_{xz}x_t + w_{hz}h_{t-1} + b_z) \tag{4.4}$$

$$\tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{4.5}$$

At the same time the input gate is calculated. This gate decides what should be added to the cell state (c_t) from the current input (x_t), the previous hidden state

(h_{t-1}), and the previous cell state (c_{t-1}), as shown in Equation 4.6.

$$i = \sigma(w_{xi}x_t + w_{ci}c_{t-1} + b_i) \tag{4.6}$$

The sigmoid activation function of the input gate (i) outputs a value between zero and one. Therefore, when the block input and input gate are multiplied together, only the values the input gate has determined important are passed through. The current cell state can be calculated as $z \times i$.

The second gate calculated is the forget gate. The forget gate decides what part of the cell state should be allowed to pass through and what should be forgotten from the current input (x_t) and the previous hidden state (h_{t-1}), as shown in Equation 4.7. The output of the forget gate enters the CEC and is multiplied by the previous cell state (c_{t-1}), then added to the output of the previous step, block input and input gate, as shown in Equation 4.8.

$$f = \sigma(w_{xf}x_t + w_{cf}c_{t-1} + b_f) \tag{4.7}$$

$$c_t = c_{t-1} \circ f_t + z \circ i \tag{4.8}$$

Gers, Schmidhuber, and Cummins would go on to add *peepholes* to the architecture [103] which allows the gates to connect to the CEC. Therefore, the hidden state (h_t) in Equations 4.6, 4.7, and 4.9 has been replaced with cell state (c_t).

Lastly, the output gate is calculated. The output gate decides how much of the new memory should be passed through to the output from the cell based on the previous hidden state (h_{t-1}) and the current input data (x_t), as shown in Equation 4.9. Again, an activation function of sigmoid (Equation 3.4) is applied to squash the values between zero and one.

$$o = \sigma(w_{xo}x_t + w_{co}c_t + b_o) \tag{4.9}$$

The cell state is then passed through an activation function, usually tanh, to regulate its values between minus one and one, then multiplied by the output gate's output to determine the output of the cell, and what will be passed forward as shown in Equation 4.10.

$$h_t = tanh(c_t) \circ o_t \tag{4.10}$$

GRU Model

The gates included in a standard GRU cell are an *update gate* and a *forget gate*, as shown in Figure 4.3. The current input (x_t) and the previous hidden state (h_{t-1})

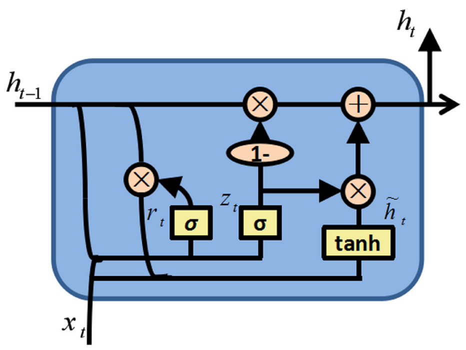

FIGURE 4.3: A gated recurrent cell [34]

is added together and passes through the update gate, as shown in Equation 4.11. The update gate decides what information should be forgotten and what should be added. A sigmoid activation function (Equation 3.4) is used to squash the values of the input between zero and one, as shown in Equation 4.11.

$$u = \sigma(w_{xu}x_t + w_{hu}h_{t-1} + b_u) \tag{4.11}$$

Next, the same input (x_t and h_{t-1}) is passed through the reset gate with a sigmoid activation function (Equation 3.4), as shown in Equation 4.12. The reset gate is used to decides how much of the past information should be forgotten.

$$r = \sigma(w_{xr}x_t + w_{hr}h_{t-1} + b_r) \tag{4.12}$$

The hidden state (h) is then updated using the reset gate and the current input (x_t) (as shown in Equation 4.13), where the product of the reset gate (r_t) and the weighted previous hidden state ($w_{hh}h_{t-1}$) is the *Hadamard* product.

$$h_t = tanh(w_{xh}x_t + (1 - r_t) \circ w_{hh}h_{t-1} + b_h) \tag{4.13}$$

Finally, the hidden state is updated using the update gate to determine what information from the current memory should be stored, as shown in Equation 4.14.

$$h_t = z_t \circ h_{t-1} + (1 - z_t) \circ h_t \tag{4.14}$$

4.3.3 Performance Metrics

In order to evaluate and compare the accuracy of the models the RMSE, as shown in Equation 2.3, was used to measure the average deviation between the predicted value and the actual value.

The RMSE is a standard performance metric for the evaluation of time series prediction models; it is also the most commonly used in road traffic flow prediction [44]. However, the fundamental problem with this performance metric, and its counterparts such as mean absolute percentage error, is they only evaluate the accuracy of a model. No consideration is given to the training time, or the trade-off between accuracy and training time. This is not appropriate for DNNs. As technology advances, so does the depth and breadth of DNNs that are feasible. However, these advancements come at a price, the training time. DNNs must be assessed on their overall performance, accuracy, and training time. To compound the issue further, limited studies that do consider training time find it challenging to compare the accuracy and training time due to both results being in different units of measurement. Therefore, performance metrics like RMSE for DNNs are antiquated.

We propose a novel performance metric, the **St**andardised **A**ccuracy and **T**ime **S**core (STATS), as shown in Equation 18. STATS overcomes the limitations of the standard performance metric by evaluating both the accuracy and the training time of a model. Furthermore, these evaluations are standardised in a set range (0 to 100, where the higher the score the better the performance) allowing comparisons to be made. Additionally, due to the scores being standardised they can be combined (accuracy and training time) to give a model an overall performance score.

The accuracy and training time scores are normalised by their ratio to the range of the results, as shown in Equation 4.15. The range can be adapted for specific time series problems through the use of upper and lower bounds, as shown in Equation 4.16 to 4.18. The bounds may be used to disregard any accuracy or training times which fall beyond a set range determined by the user. For example, setting an upper bound of 20 for the accuracy score will score any model with an RMSE of 20% or above as zero for accuracy, and vice versa, setting a lower bound on accuracy as 10%, any models achieving 10% or better will be scored as full marks (100%) for accuracy. The same logic extends to the upper and lower bounds of the training times.

$$STATS = \left(\frac{w_a}{\alpha_a - \beta_a} \times (\alpha_a - x_a)\right) + \left(\frac{w_t}{\alpha_t - \beta_t} \times (\alpha_t - x_t)\right) \tag{4.15}$$

$$x_a \equiv x_a(s) = \begin{cases} U & \text{if} \quad s \geq U, \\ L & \text{if} \quad s \leq L, \\ s & \text{otherwise} \end{cases} \tag{4.16}$$

$$\alpha_\alpha \equiv \alpha_\alpha(s) = \begin{cases} U & \text{if} \quad s \geq U, \\ s & \text{otherwise} \end{cases} \tag{4.17}$$

$$\beta_a \equiv \beta_a(s) = \begin{cases} L & \text{if} \quad s \leq L, \\ s & \text{otherwise.} \end{cases} \tag{4.18}$$

The equations (4.16–4.18) hold for x_t, α_t, β_t respectively where α_a is the largest accuracy (RMSE), β_a is the smallest accuracy (RMSE), α_t is the largest training time, β_t is the smallest training time, w_a is the weighting for the accuracy score, w_t is the weighting for the training time score (where $w_a + w_t = 100$), x_a is the actual accuracy (RMSE), and x_t is the actual training time. In this chapter, there were no external constraints to take into consideration during experimentation. Therefore, no upper or lower bounds were used and a priori equal weighting was given to the accuracy and training time (wa = wb = 50) to express lack of specific weighted preference..

4.3.4 Comparison of Existing Deep RNNs

To answer research questions one and two, what is the most suitable deep RNN for road traffic flow prediction based on its accuracy and training time and what is the most suitable performance metric for DNNs, three deep RNNs were used with the 12 different architectural structures, as described in Section 4.3.2, and applied to the real traffic flow dataset, as illustrated in Section 4.3.1. The models were used to predict the traffic flow at the next time horizon (five minutes). A comparison of the models was made based on their accuracy using the performance metric RMSE (Equation 2.3) and their accuracy, training time, and overall score using the novel performance metric, STATS, as shown in Equation 4.15. The RMSE and STATS results were then compared.

TABLE 4.1: The comparison of the performance scores of the deep recurrent neural network models

Model	**RMSE (%)**	**STATS**		
		Accuracy Score (%)	**Time Score (%)**	**Overall Score (%)**
RNN	17.32	80.46	98.79	89.62
LSTM	9.71	98.32	82.23	90.27
GRU	9.26	98.59	82.77	90.68

Table 4.1 shows the results of the model architecture that provided the best RMSE and the best overall performance score using STATS, with a breakdown of its accuracy and time score, for each deep RNN. In terms of the RMSE, the GRU was the most accurate with an RMSE of 9.26%, closely followed by the LSTM with an RMSE of 9.71%. The RNN model was the least accurate with an RMSE of 17.31%. Based on the RMSE it could be assumed that the GRU and LSTM's performances are of comparable success. Table 4.1 also shows that, overall, the GRU was the most successful model for the STATS performance metric. It was the most accurate, with an accuracy score of 98.59%, again, closely followed by the LSTM, with an accuracy score of 98.32%. The RNN was the least accurate model with an accuracy score of 80.46%. However, the STATS also shows that the GRU is faster than the LSTM, with a time score of 82.77% and 82.23% respectively, providing more insight into the model's performance.

The STATS performance metric was able to provide additional information when compared to the RMSE. The RMSE evaluated the models solely on their accuracy. No consideration was given to the training time, or the trade-off between accuracy and training time. The STATS assessed the models on their accuracy and training time, allowing for a more balanced evaluation of the deep learning model. It clearly shows the trade-off between accuracy and training time, as demonstrated by the RNN model, enabling a more informed selection of which model is superior for road traffic flow prediction.

4.3.5 Comparison of Existing Deep RNNs' Sensitivity to Architectural Change

To answer research question three, what is the most suitable DNN (RNN) for road traffic flow prediction based on sensitivity to architectural change (the model's range of predictions over various architectural structures), three deep RNNs were used with the 12 different architectural structures, as described in Section 4.3.2, and applied to the real traffic flow dataset, as illustrated in Section 4.3.1. The models were used to predict the traffic flow at the next time horizon (five minutes). A comparison of the models was made based on their accuracy, training time, and overall performance using the novel performance metric, STATS, as shown in Equation 4.15. Additionally, the STATS results were analysed (range, mean, and standard deviation) to determine which model was most sensitive to architectural change (the model's range of predictions over various architectural structures).

TABLE 4.2: The performance scores of the deep recurrent neural network model

Performance Indicator	**STATS**		
	Accuracy Score (%)	**Time Score (%)**	**Overall Score (%)**
Highest	80.46	100	89.62
Lowest	29.42	68.36	50.46
Mean	55.61	84.55	70.08
Range	51.05	31.61	39.16
Standard Deviation	13.72	12.19	12.12

Table 4.2 shows that the RNN architectures were consistently fast, with the highest mean time score of 84.55%, a small range of 31.61%, and a small standard deviation of 12.19%. This is due to the RNN computational structure, as shown in Figure 4.2, which allows the model to be computational efficient at the price of poor prediction results. The accuracy score was less stable, with a mean score of 55.61%, a range of 51.05%, and a standard deviation of 13.72%. The model's accuracy was sensitive to architectural structure changes. Therefore, the RNN performed the worst overall. The simplistic RNN computational structure is unable to capture the long-term dependencies within time series data. It is unable to learn the dependence between input and output if the time lag is greater than 5-10 discrete timesteps. This has limited the contextual information that the RNN can use for prediction. The results show that more contextual information from distant time lags is needed to produce successful prediction results. Therefore, an RNN is not suitable for the prediction of road traffic flow.

TABLE 4.3: The performance scores of the deep long short-term memory neural network model

Performance Indicator	STATS		
	Accuracy Score (%)	Time Score (%)	Overall Score (%)
Highest	98.32	83.18	90.57
Lowest	0	0	40.48
Mean	78.28	47.23	62.78
Range	98.31	83.18	49.45
Standard Deviation	27.19	29.25	17.10

Table 4.3 shows that the LSTM's accuracy and time score were both very unstable. The model produced the largest range and standard deviation for both the accuracy and time score of all the models. Therefore, both the model's accuracy and training time is highly sensitive to architectural structure change. This can be attributed to the intricate structure of an LSTM cell, as shown in Figure 4.1. The CEC allows the model to link cause and effect over many distant time lags, leading to an improvement in prediction accuracy over the RNN. However, the cell contains multiple gates. Each gate is a single-layer neural network. Multiple neural networks have caused the model's training times to be slow and very sensitive to architectural change. Therefore, using an LSTM is a trade-off between accuracy and training time.

TABLE 4.4: The performance scores of the gated recurrent unit neural network model

Performance Indicator	STATS		
	Accuracy Score (%)	Time Score (%)	Overall Score (%)
Highest	100	83.31	90.68
Lowest	55.71	16.55	39.80
Mean	87.22	51.03	69.12
Range	44.29	66.76	50.88
Standard Deviation	13.18	25.22	17.11

Table 4.4 shows that the GRU produced the highest mean accuracy score, with the lowest range and lowest standard deviation. Therefore, the GRU's accuracy was the most stable throughout the experimentation, and so was the model least sensitive to structural changes. The GRU's time score range and standard deviation were larger than the RNN's, however, its mean, range, and standard deviation were better

than the LSTM. This is to be expected. The time scores will only increase/decrease exponentially with the number of layers or nodes added/removed for all models. Therefore, when determining which model was overall the least sensitive to architectural change, accuracy was the determining factor.

Consequently, the GRU was overall the least sensitive to architectural change. The GRU's stable accuracy is due to the intricacy of its cells, as shown in Figure 4.3, which contains an internal memory that allows the model to learn dependencies over many time lags, more than a standard RNN, adding more contextual information to its predictions. However, what has helped the GRU, has hindered the LSTM. The LSTM is computationally more complex than the GRU and contains three gates. A forget gate (f), an input gate (i), and an output gate (o). Each gate is a single neural network. It also contains two cell states, a hidden state (h) and a memory state (C), to retain past information (as shown in Figure 4.1). This complex structure produces good results when the architecture is calibrated correctly. However, when not properly constructed, it can lead to overfitting and therefore, poor generalisation when testing.

4.4 Summary

In this chapter, we have performed a comparative analysis of three deep RNN architectures (an RNN, an LSTM, and a GRU) by applying them to a real dataset to determine which was most successful for the prediction of road traffic flow based on their accuracy and training time. We also developed a novel performance metric, STATS, that allowed the comparison of both accuracy and training time by standardising them into a comparable score for the overall evaluation of DNNs. Furthermore, we then compared the DNNs' sensitivity to architectural changes to determine how stable the models' prediction accuracy were. The results showed that the GRU model produced the most overall successful prediction, producing good results in both accuracy and training time. Furthermore, the GRU model was the least sensitive to architectural change. Therefore, based on these results it can be concluded that the long-term patterns embedded within the input data are needed for the successful prediction of road traffic flow.

Based on these experimental results, if the long-term temporal patterns embedded in the data are crucial for the prediction of road traffic flow, it should be asked which long-term temporal patterns are important? A day, a week, a month, or a year? Furthermore, as a prediction model can only be as good as its input data, how can such long-term temporal patterns be inputted into a prediction model? These issues are complicated further by online learning. The models in this work are statically trained, however, to have a real-life application they will need to advance to dynamic/online learning models. The main issue with online learning is catastrophic forgetting. Due to the continual updating of an ANN's weights and biases based on the most recent data point(s) the ANN will eventually converge to the short-term temporal patterns, forgetting previously learnt long-term temporal patterns. Therefore, different magnitudes of temporal patterns, such as day, week, month, and year, should be examined to ascertain how contextual temporal data can improve road traffic prediction. Furthermore, a framework should be developed that can overcome catastrophic forgetting and learn temporal patterns dynamically.

Chapter 5

A Novel Online Dynamic Temporal Context Neural Network Framework for the Prediction of Road Traffic Flow

This chapter is a modified version of 'A Novel Online Dynamic Temporal Context Neural Network Framework for the Prediction of Road Traffic Flow' published in IEEE Access, 7, 2019, and has been reproduced here with the permission of the copyright holder.

Chapter Summary

In the previous chapter, we compared three deep RNN architectures. The results, summarised in Section 4.4, highlighted the importance of long-term temporal patterns within the data for road traffic flow prediction. Therefore, in Section 5.4.4 we investigate the different magnitudes of temporal patterns, detailed in Section 5.4.1, to understand how contextual temporal data can improve prediction. Furthermore, in Section 5.3.1 we propose a novel online dynamic temporal context neural network framework that overcomes catastrophic forgetting and can learn temporal patterns dynamically. In Section 5.4.4 we evaluate the proposed framework's performance.

5.1 Introduction

Existing work into road traffic prediction has focused on using small training datasets, ranging from a few days to a few weeks [33] [38]. However, a prediction model can only be as good as its input data [40]. The temporal magnitude of the training data will determine and restrict what temporal cycles and patterns can be learnt. Despite this weakness, past research has neglected to investigate what temporal patterns are important and should be included within the training dataset. Most assume only short-term patterns, such as hourly and daily, are needed based on no prior investigations [33] [38]. Research by Williams and Hoel [16] has shown that traffic flow in urbanised areas does exhibit weekly patterns linked to the working week, however, other temporal patterns are important. Traffic flow in urbanised areas also exhibits long-term patterns, such as monthly and even yearly. These patterns include, but are not limited to, less traffic during the summer months and increased traffic in December and January. Therefore, the inclusion of short-term and long-term patterns within the training data could improve prediction results.

Furthermore, DNNs, especially in the traffic flow prediction field, are traditionally statically (not incrementally or online) trained [111] [37] [112] [113]. Therefore, the learning capacity of these models is restricted to patterns and events that occurred during the training dataset, such as recurring traffic congestion. This is impractical for real-life applications. Road traffic flow data is complex and stochastic [68], hence, their prediction models must be able to adapt to previously unseen events, such as non-recurring road traffic congestion or a road traffic incident. One way to overcome this problem is to use online learning. Online learning is a machine learning approach that uses the most recent sequential data point or points to update the model's weights and biases as soon as the data is available. This can improve the prediction accuracy of complex and stochastic sequential data, such as road traffic flow. However, online learning does have its limitations. The main disadvantage of online learning is the eventual loss of the long-term temporal patterns embedded within the training data. By continually updating the DNN's weights and biases based on the most recent data point or points, the model will eventually converge to the short-term temporal patterns, forgetting previously learnt long-term temporal patterns. This is known as catastrophic forgetting. Therefore, research into DNN's architectures that can learn and retain short and long-term temporal patterns during online learning needs to be investigated further.

The contributions and novelty of this chapter include:

1. we have investigated different magnitudes of temporal patterns (long and short-term), through the use of different temporal data segments to understand how contextual temporal data can improve prediction; and

2. we have developed a novel online dynamic temporal context neural network framework. The framework uses different temporal data segments as input features, and during online learning, the updating scheme can dynamically determine how useful different temporal data segments are and weight them accordingly for use in the regression model. Therefore, the model can include relevant long-term temporal patterns in the regression model leading to improved prediction results.

5.2 State-of-the-Art in Deep Convolutional Neural Networks for Road Traffic Flow Prediction Models

In Section 4.2, a review of deep RNNs was performed. Although it highlighted that deep RNNs provide good prediction results [30] [114], they can be computationally costly when processing high dimensional data [115]. Therefore, in this section we move on to explore another deep ANN structure designed to process high-dimensional data, a Convolutional Neural Network (CNN) [115].

A CNN [35] is a feed-forward neural network that uses the geographical proximity of its input data points to add a geospatial dimension to the prediction function being learnt. Consequently, CNNs are traditionally used when the input data can be expressed in terms of a map, such as an image analysis. Nevertheless, many other data sources possess similar characteristics. CNNs combined with RNNs have been used in image/text analysis experiments such as Peris et al. [116], Wang et al. [36], and Lopez-Martin [117]. This research has paved the way for CNNs to be used for road traffic flow prediction.

Narmadha and Vijayakumar [118] developed a hybrid prediction model containing two 1D CNN layers to capture the spatial dependencies and two LSTM layers to capture the temporal dependencies. The proposed model used three months of data, from Performance Measurement Systems (PEMS), split into two months for training and one month for testing. The data contained a time horizon of five minutes and input features of traffic flow and weather details from three RTSSs (the site of prediction and its up and downstream neighbours). The proposed model was compared to five other models; 1) ARIMA, 2) KNN, 3) LSTM (univariate), 4) LSTM (multivariate) and 5) a CNN (multivariate). The experimental results show that the CNN-LSTM hybrid model was the most successful model. However, there are a number of issues with the study. Firstly, it only considered a LSTM RNN. A GRU RNN, as shown in Section 4.3.4, may have been more successful. Secondly, the model was statically trained, therefore it has assumed that the temporal and spatial relationships between the input features are constant. Furthermore, it can not cope with abnormal traffic flow that has not been seen in the training data. Thirdly, it has assumed that the only temporal patterns significant for road traffic flow prediction are daily and weekly.

Wu et al. [37] built upon the research by Wang et al.[36] and also developed hybrid model to predict road traffic flow. Two GRU layers were used to detect temporal features and three CNN layers were used to detect spatial features were run concurrently. Their outputs were combined into a single regression layer to make a prediction. Additionally, to detect patterns across different time lags, three different segments of historical input data (all 105 minutes in length) were used. The segments were from: 1) immediately preceding the prediction, 2) exactly one day before the prediction, and 3) exactly one week before the prediction. The input segments were also preprocessed in an attention model before entering the RNN or CNN layers. Three months of data from 33 RTSSs were used to train and test the model to predict multiple time horizons of five minutes. Its results were compared to five state-of-the-art time series prediction models, and Wu et al. determined that the GRU and CNN hybrid was the most accurate. However, assumptions are made over the temporal segments. It has been assumed that only the daily and weekly temporal patterns are significant; no consideration was given to monthly or yearly patterns. Furthermore, again the model was only trained statically, it has assumed that the relationship between the temporal data segments is constant. Once the model has learnt the temporal and spatial relationships contained within the training data it has no opportunity to update these relationships based on the current data. Therefore, it does not lend itself to real-life applications such as road traffic incidents. A model which includes online learning would be more appropriate.

In conclusion, CNNs are still in their infancy in terms of application for road traffic flow prediction [119]. Many papers exploring architecture hybrids within image

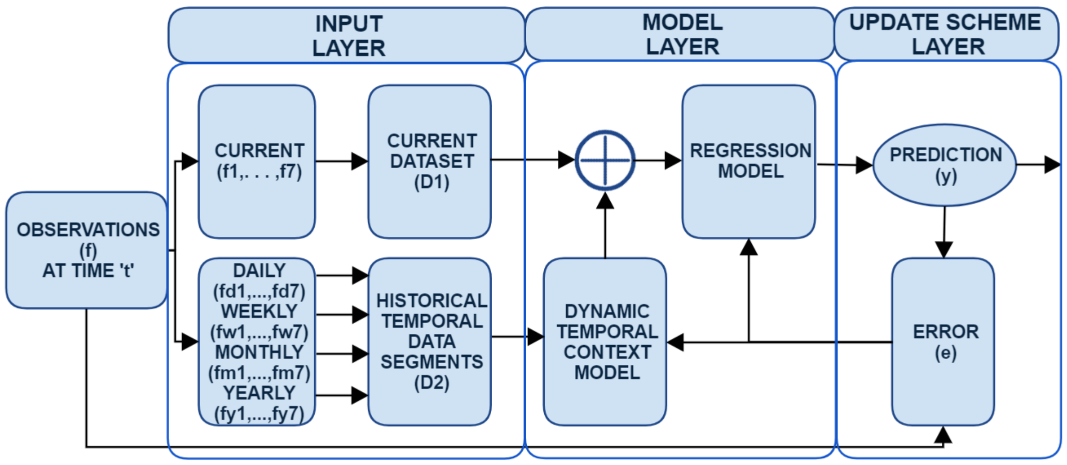

FIGURE 5.1: The proposed framework

analysis and text/speech analysis have started to cross over into time series prediction, however, one major hurdle that needs to be overcome for CNNs to make a significant impact on time series prediction is its ability to detect short and long-term patterns embedded within the data. Furthermore, another issue highlighted by the literature review is the lack of consensus over what magnitude of temporal data should be used, or if providing historical temporal data from distant time lags can provide context and improve prediction accuracy. Most research fails to address the temporal element of input data. The limited research that does address the temporal element does not compare their model with and without the addition of the temporal data to assess its impact on the model's accuracy [108]. Furthermore, the additional temporal data is often chosen through expanding the current temporal dataset [120], with no justification and may be irrelevant [37]. Banko and Brill [40] identified that the input data used was the most important element of a successful machine learning model. Therefore, further research into input data for DNNs and their temporal magnitude is vital.

5.3 Methodology

5.3.1 The Proposed Framework

We have developed a novel online dynamic temporal context (DTC) neural network framework, as shown in Figure 5.1. The framework uses different temporal data segments as input features, and, during online learning the updating scheme can dynamically determine how useful different temporal data segments are for prediction accuracy. The different temporal data segments are then weighted according to their usefulness for the regression model and added to the current observations. Therefore, the framework can include short and relevant long-term temporal patterns in the regression model leading to improved prediction results.

The framework can be divided into three distinct components: 1) an input layer, 2) the model layer, and 3) the update scheme layer, as seen in Fig. 5.1. Each layer will now be defined in more detail.

The Input Data Layer

Unlike traditional regression neural networks, the proposed framework has two sources of input data: 1) the current observations (D_1), and 2) the corresponding different temporal data segments (D_2).

The current observations (D_1) are the traffic flow observed immediately before the prediction point ($t + 1$). The current observations dataset is a 7d array, as shown in Equation 5.1, containing the total traffic flow and its breakdown into six different vehicle classes, as shown in Table 5.1. Vehicle classes are used as input features (f) for both the DTC model and regression model based on prior research which demonstrated that vehicle classes can improve prediction results [121].

$$D_1 = \begin{pmatrix} f_{1,t} & f_{2,t} & \dots & f_{n,t} \\ f_{1,t-1} & f_{2,t-1} & \dots & f_{n,t-1} \\ f_{1,t-2} & f_{2,t-2} & \dots & f_{n,t-2} \\ \dots & \dots & \dots & \dots \\ f_{1,t-n} & f_{2,t-n} & \dots & f_{n,t-n} \end{pmatrix} \tag{5.1}$$

TABLE 5.1: An extract from the current (t) traffic flow observations (D_1)

Total	Total	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
t	147	12	123	2	9	2	1
t_{-1}	139	10	115	0	10	1	3
t_{-2}	142	9	117	1	9	2	3
t_{-3}	148	8	119	3	11	0	7
...	12	1	8	0	2	1	0
t_{-n}	58	3	51	0	4	0	0

The different temporal data segments (D_2) are the corresponding observed traffic flow data that is one day, one week, one month, and one year before the prediction point ($t + 1$). Each temporal data segment is a 7d array containing seven different features ($f_i \Rightarrow i \in \mathbb{Z} : 1 \leq i \geq 7$). This includes the total traffic flow and its breakdown into six different vehicle classes matching the current observations' shape and structure, as shown in Equation 5.1. In total, the different temporal data segments dataset is a 28d array, as shown in Equation 5.2, where d denotes daily, w denotes weekly, m denotes monthly, and y denotes yearly data segment.

$$D_2 = \begin{pmatrix} f_{[d_1,d_n],t} & f_{[w_1,w_n],t} & f_{[m_1,m_n],t} & f_{[y_1,y_n],t} \\ f_{[d_1,d_n],t-1} & f_{[w_1,w_n],t-1} & f_{[m_1,m_n],t-1} & f_{[y_1,y_n],t-1} \\ f_{[d_1,d_n],t-2} & f_{[w_1,w_n],t-2} & f_{[m_1,m_n],t-2} & f_{[y_1,y_n],t-2} \\ \cdots & \cdots & \cdots & \cdots \\ f_{[d_1,d_n],t-n} & f_{[w_1,w_n],t-n} & f_{[m_1,m_n],t-n} & f_{[y_1,y_n],t-n} \end{pmatrix} \tag{5.2}$$

Both sources of input data, current observations and different data segments (D_1 and D_2), are passed to the model layer for processing.

The Model Layer

The model layer contains two models with different architectures: 1) the DTC model architecture, and 2) the regression (GRU) model architecture.

The proposed DTC model has a CNN structure. Traditionally, CNN structures are used for static tasks where input data can be expressed in terms of a map, such as an image analysis or classification. In addition, cutting edge research into time series prediction has used CNN to find geospatial relationships between different geographical locations to help improve prediction accuracy. Our proposed model is different from previous time series prediction models using CNNs as we seek to find relationships between different magnitudes of temporal data segments. The model uses the different temporal data segments (D_2) to dynamically determine how useful it is for the regression model (GRU) to produce an accurate prediction. It does this by weighting the input segments. What differentiates the proposed model from traditional CNN architectures is; 1) we have used temporal data as input features (f_i), therefore, in the proposed model the kernel scrolls 'across' the temporal data segments (D_2) and not down the temporal data like traditional arrangements of CNNs,

2) the kernel (k) used to detect temporal patterns is rectangular and not square as traditionally used in CNNs, so the kernel (k) only convolves across one line of input data at once, 3) the model uses downsampling to obtain the most relevant temporal data, therefore, no padding function is used unlike in traditional CNN structures to maintain the dimensions of the input data, and 4) the stride (s) used for the kernel (k) is equal to the width of the kernel ($k = s$) to ensure that each data point is only convolved over once by the kernel (k) per layer (ℓ). This enables the DTC to reduce the dimensionality of the input data while ensuring no replications are passed on to the regression model. The DTC model will now be defined in more detail.

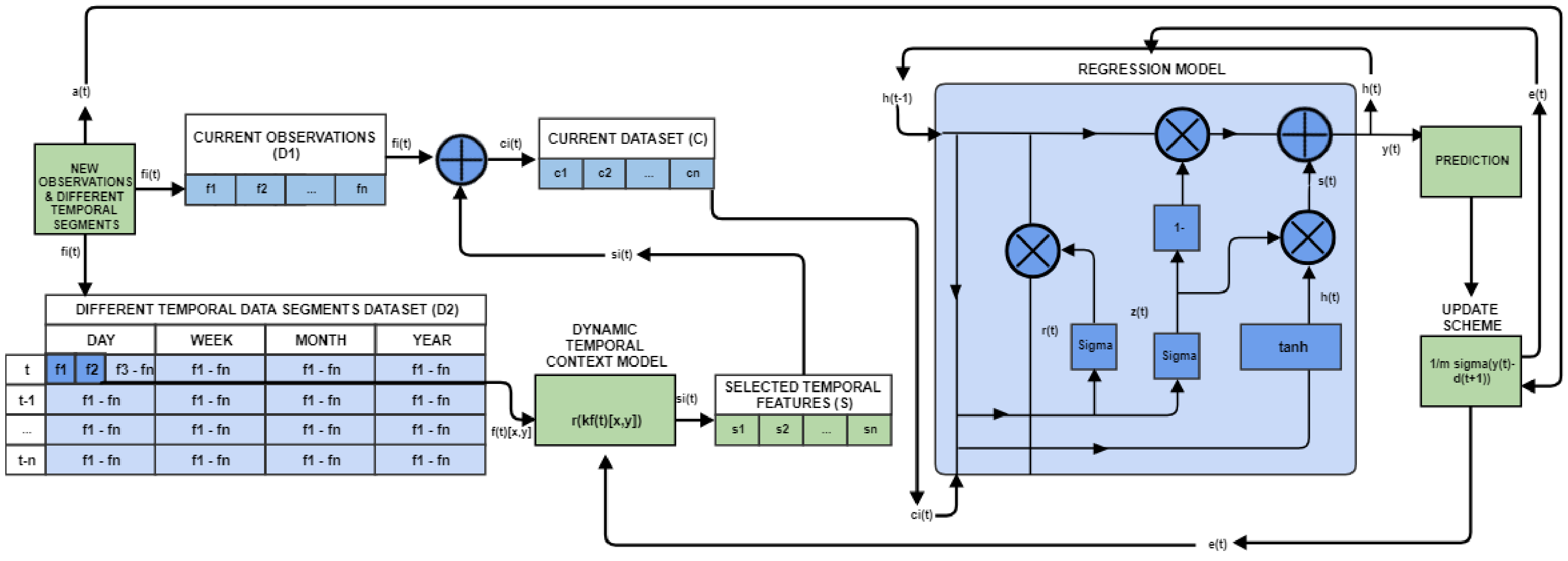

FIGURE 5.2: The proposed Dynamic Temporal Context framework

The proposed DTC model's input is the 28d array of different temporal data segments (D_2); its structure is a CNN, as shown in Equation 5.2. In the convolutional layer a convolution kernel (k), also known as a filter or feature detector, convolves (slides) over the different temporal data segments (D_2) input features (f_i) until every input feature has been passed over, moving left to right. Therefore, temporal data is used as an input feature in the array columns and rows, contrary to traditional CNN structures. The convolutional operation ($k[x, y]$), where x and y define the current position of the kernel (k) in the dataset D_2, can be defined as

$$kD_2 = k \otimes f_i : f_i \in D_2[x, y] \tag{5.3}$$

In the proposed model the magnitude of the movement made to the right is known as a *stride* (s) and is defined as the same length as the convolutional kernel (k), therefore, $s = k$, and is a rectangle, unlike traditional CNN kernels. This constraint has been set to ensure each feature (f_i) is passed over only once in each layer (ℓ) per kernel (k) to ensure that the output contains no duplication. At each stride (s) the weights (w_i) in the kernel (k) are multiplied by the corresponding indices ($d \in D_2$) position (x and y) underneath in the temporal segments data ($k \otimes d$) to create the *convolution*. The calculated values are used to create one output value, as shown in Equation 5.3, and used to construct the *feature map* (M), as shown in Figure 5.2. What is considered an important temporal pattern by the proposed model is learned during the training process. Multiple kernels (k) can be used to detect multiple important temporal patterns in the temporal data segments. Every hidden layer (ℓ_h) has at least one kernel (k), and the depth of the feature map (M) is determined by the number of kernels in the hidden layer (ℓ_h). The number of kernels (k) and hidden layers (ℓ_h) the DTC contained was optimised through grid search.

It should be noted that although the literature refers to the above process as a convolution, technically the implementation in the proposed model, and most other implementations of CNNs, used a *correlation operation*. Both operations are closely related, with both being neighbourhood operations. The only significant difference between the two operations is during the calculation of a convolution the kernel (k) is rotated 180 degrees; the kernel (k) does not rotate during the correlation calculation. Therefore, for clarification, in the paper when referring to the convolution operation of our proposed model, we are referring to a correlation operation.

The convolutional operation is linear, therefore, an *activation layer* (ℓ_a) follows the convolutional layer to account for the non-linear relationship between the data points. In the proposed model a rectified linear unit (ReLU), as seen in Equation 5.4, activation function was used.

$$r(m) = MAX(0, m) : m \in M \tag{5.4}$$

A ReLU was used to normalise the output of the DTC between the range of $0 - x$, to ensure that none of the temporal data segments would be negatively weighted. The feature map (M) is then fed the activation layer (ℓ_a); a ReLU function (r) was applied to each data point (m) in the feature map (M) matrix to transform the data into the set range. The output of the activation layer (ℓ_a), the *activation map* (A), contains the same dimensions as its input, the feature map (M). The activation map (A) is then fed into the pooling layer (ℓ_p). The pooling layer (ℓ_p) is used to condense the temporal data segments while preserving the important temporal patterns (features (f)). A sliding window is used to move across the activation map (A), and one value is chosen per stride (s), as shown in Figure 5.2. Again, the stride is equal to the size of the window ($s = k$) to ensure no duplication in the output. Therefore, the activation map (A) is downsampled and reduced in width, to a width of q_p, as shown in Equation 5.5, where q_a is the width of the activation layers (ℓ_a) input. The value chosen in the sliding window is the largest value (*max pooling*).

$$q_p = \frac{q_a - k}{s} + 1 \tag{5.5}$$

Traditionally, the output of the pooling layer (ℓ_p) is calculated as

$$o = \frac{q - K + 2P}{s} + 1 \tag{5.6}$$

where p represents a padding function added to increase the dimensions of the output data back to its original magnitude. However, as downsampling was the aim of the proposed model, no padding function (p) was used in the proposed model.

Different from the existing time series models using CNN where the prediction models are based on static data, our proposed DTC model is dynamic and seeks to find a relationship between different magnitudes of temporal data segments promptly. In the proposed DTC model, the output is the most relevant temporal features (S) for prediction. The selected temporal features (S) are then added to the current observations (D_1) to create the current dataset (C) and passed through to the regression model (GRU), as shown in Figure 5.2. Based on previous research [122] the regression layer used was a deep GRU model. A GRU model works through the use of gates; each gate is a neural network. The gates included in a standard GRU cell are an *update gate* and a *forget gate*, as shown in Figure 5.2. The current input ($c_t \in C$) and the previous hidden state (h_{t-1}) is added together and passes through the update gate, as shown in Equation 4.11. The update gate decides what data should be forgotten and what should be added. A Sigmoid activation function is used to squash the values of the input between zero and one, where b is the bias. Next, the same input (c_t and h_{t-1}) is passed through the reset gate with a Sigmoid activation function (as shown in Equation 4.12). The reset gate is used to decides how much of the past information should be forgotten, as shown in Figure 5.2. The hidden state (h) is then updated using the reset gate and the current input (c_t) (as shown in Equation 4.13), where the product of the reset gate (r_t) and the weighted previous hidden state ($w_{hh}h_{t-1}$) is the *Hadamard* product. Finally, the hidden state is updated using the update gate to determine what information from the current memory should be stored, as shown in Equation 4.14. The output then predicts the number of vehicle (y_t) at the next time point ($t+1$), as shown in Figure 5.2. Once the regression model, GRU, has made its first prediction (y_t) using the test data, the prediction (y_t) and the actual value (a_{t+1}) are then passed to the Update Scheme layer, as shown in Figure 5.1.

The Update Scheme Layer

The primary objectives of the Update Scheme layer are: 1) to update the weights and biases in the DTC model to dynamically and timely adjust the most relevant temporal features from the temporal data segments dataset (D_2) for use in the regression model, and 2) to update the weights and biases in the GRU model to allow the model to adjust and adapt to changing temporal trends within the time series data. This was done through online learning. Once a prediction (y_t) has been made, the actual value (a_t) is added as a new line of observations to the current observations dataset (D_1) and its corresponding temporal data segments are added to D_2, as shown in Figure 5.2. The prediction (y_t) and actual observation (a_t) are then compared, and its error ($\epsilon = y_t - a_t$) is computed and passed back to the DTC model. This is done to update the model's weight (w_i) and biases (b_i) contained within the kernels (k_i) to allow the model to dynamically adjust the most relevant temporal data segments for regression based on the most recent time series data. This is achieved through the use of a stochastic gradient descent method [93] and a small window of the most recent data segments in dataset D_2. During backpropagation, using a small window of the most recent data in D_2, the gradient of the error (ϵ) is found with respect to the DTC model's weights (w_i) and biases (b_i) using differentiation, as seen in Equation 5.7.

$$\frac{\delta\epsilon}{\delta w_i} \quad \text{and} \quad \frac{\delta\epsilon}{\delta b_i} \tag{5.7}$$

The error's (ϵ) gradient is then backpropagated through the model, from the output layer (ℓ_o) to the input layer (ℓ_i), to find the global minima. In each layer (ℓ) the gradient is scaled by a learning rate (l) as shown in Equation 5.8.

$$w_{i,t} = w_{i,t-1} - l\frac{\delta\epsilon}{\delta w_i} \quad \text{and} \quad b_{i,t} = b_{i,t-1} - l\frac{\delta\epsilon}{\delta b_i} \tag{5.8}$$

The weights (w_i) and biases (b_i) in the kernel (k_i) within the DTC model are then updated accordingly to minimise the error (ϵ). Once the DTC model is updated, the new temporal features are selected ($s_{1,t+1} - s_{n,t+1}$) and added to new current observations (D_1) to create an updated current dataset (C), as shown in Figure 5.2. A window of the new current dataset (C), is then fed to the regression model (GRU) to update the weights (w_i) and biases (b_i) in the GRU layers. The regression model is updated to improve the prediction accuracy of the overall model by adapting to temporal trends within the time series data.

The regression model is also updated using stochastic gradient descent method [93]. The current input ($c_{t+1} \in C$) and the previous hidden state (h_t) is added together and passed through the update gate, as shown in Equation 4.11. The GRU cell processes the input as described in Equation 4.11 to 4.14, and the gradient of the error (ϵ) is found with respect to the regression model's weights (w_i) and biases (b_i) using differentiation, as seen in Equation 5.7. The error's (ϵ) gradient is, again, backpropagated through the regression model, from the output layer (ℓ_o) to the input layer (ℓ_i), to find the global minima. In each layer (ℓ) the gradient is scaled by a learning rate (l) as shown in Equation 5.8. The weights (w_i) and biases (b_i) within the regression model are then updated accordingly to minimise the error (ϵ). Once the Updating Scheme has updated the regression model, a new prediction is made (y_{t+1}) and the cycle continues.

5.4 Experimental Evaluation

In this section, we have focused on two research questions: 1) how do different temporal data segments affect prediction accuracy? 2) can a dynamic temporal context framework that can include both short-term and relevant long-term temporal patterns improve prediction accuracy?

5.4.1 Data Description

Both the proposed dynamic temporal context and the deep gated recurrent unit model were applied to an existing real dataset collected from a typical busy urbanised arterial road between Manchester and Liverpool, UK. The dataset consisted of three months of data collected from 1st January to 31st March 2016, with a time horizon of five minutes (26,195 data points). Historic datasets, referred to as temporal data segments, were added as input features to give the data temporal context. The temporal data segments added to the original dataset were the previous day, week, month, and year, as shown in Table 5.2.

TABLE 5.2: The temporal datasets

Dataset	Description
1	Current dataset with no temporal data segments
2	Current dataset with previous day temporal data segment
3	Current dataset with previous week temporal data segment
4	Current dataset with previous month temporal data segment
5	Current dataset with previous year temporal data segment
6	Current dataset with all temporal data segments

All temporal data segments were three months in length, with a time horizon of five minutes, and 26,195 data points, to correspond with the original dataset. The input data also included input features of different vehicle classes, as shown in Table 3.1, as different vehicle classes have been shown to improve prediction accuracy [121].

Therefore, the total dataset contains 26,195 data points and 35 different input features. Two months of the dataset were used to train and validate the framework and one month was used for validating and testing. No data points were missing, therefore, no pre-cleaning of the data was necessary.

5.4.2 Model Architectures and Hyperparameters

There is currently no standard procedure or analytical calculation to determine the optimal structure or set up for any neural network, therefore, the architecture and hyperparameters of all neural networks used during experimentation were optimised using prior knowledge from the literature review or heuristics through grid search.

All weights and biases were randomly initialised and a dropout rate of 50% was used based on research by Zhao, Chen, Wu, *et al.* [33]. All other hyperparameters, such as the number of layers, nodes, and update window size, were determined through brute force by a random grid search to find the optimal design.

The grid search searched through different architectural structures ranging from two to six layers (excluding any input and output layers) and nodes ranging from 12 to 20 with different hyperparameters to find the optimal design. The different architectural structures and their variations were initially trained using the two months of training data, as described in Section 5.4.3. The models' weights and biases were then continually updated during the one-month test data through online learning (time series/rolling cross-validation).

The Adamax optimiser [93] was used to optimise the model's learning rate based on the data's characteristics (a separate step size, known as the learning rate, for each parameter) during both the training and test phase. Each variation of the models' architectures was trained and tested 10 times and an average empirical error, based on the performance metric described in Section 5.4.3, was taken. The architecture that produced the lowest average empirical error was used for comparison.

5.4.3 Performance Metrics

To evaluate and compare the accuracy of all the models, the Root Mean Squared Error (RMSE), as shown in Equation 2.3, and the STATS [122], as shown in Equation 4.15, will be used was used. There were no external constraints to take into consideration during experimentation. Therefore, when using the STATS performance metric no upper or lower bounds were used and a priori equal weighting was given to the accuracy and training time ($wa = wb = 50$) to express lack of specific weighted preference.

5.4.4 The Evaluation of Different Temporal Data Segments and the Proposed Dynamic Temporal Context Framework

To address research question one, how do different temporal data segments affect prediction accuracy, we have applied a deep gated recurrent unit model to six different datasets containing different temporal data segments, as shown in Table 5.2).

TABLE 5.3: The prediction accuracy of different temporal datasets using a deep gated recurrent unit model for road traffic flow

Model	Dataset	RMSE (%)	Accuracy Score	Time Score	Overall Score
GRU	**1**	14.64	0.00	100.00	57.32
GRU	**2**	13.95	28.92	95.24	62.08
GRU	**3**	13.58	36.79	85.71	61.25
GRU	**4**	14.01	26.42	90.47	58.45
GRU	**5**	14.57	3.08	80.95	42.02
GRU	**6**	13.57	44.58	14.28	29.43
DTC	**6**	12.24	100.00	0.00	50.00

Table 5.3 shows that the addition of any temporal data segment, even long-term, improved the prediction accuracy of the model. Therefore, long-term temporal patterns, such as monthly and yearly patterns, embedded within the data, have aided the prediction model.

Table 5.3 also shows that the inclusion of the weekly temporal data segment provided the most improvement to the prediction accuracy, with an RMSE and a STATS accuracy score of 13.575% and 28.92, respectively. The improvement was more than the daily temporal data segment, which had an RMSE and STATS accuracy score of 13.95% and 28.92, respectively. This will be due to the weekday and weekend split linked to the working week, which traffic flow in most urbanised areas exhibits. Furthermore, the inclusion of a yearly temporal data segment provided the least improvement to the prediction accuracy, with an RMSE and a STATS accuracy score of of 14.570% and 3.08, respectively. This will be due to issues such as concept drift, changes to the road network, and the redevelopment of urbanised areas. Interestingly , including all temporal data segments improved the prediction accuracy further, with an RMSE and an accuracy score of 13.574% and 44.58, respectively. However, this improvement in accuracy has come at a cost to the training time, with a STATS time score of 14.28. In conclusion, this shows that both short and long-term temporal patterns embedded within traffic flow data are important for the prediction and can improve prediction results, but the improvement in accuracy is a trade-off with training time.

To address research question two, can a dynamic temporal context framework that can include both short-term and relevant long-term temporal patterns improve prediction accuracy, the sixth dataset, as shown in Table 5.2, was used with the proposed dynamic temporal context framework and its prediction results were compared with a deep gated recurrent unit model. The results are shown in Table 5.3.

The proposed framework was more successful than the deep gated recurrent unit model at predicting road traffic flow using the same existing real dataset (dataset six from Table 5.2), with an RMSE of 12.244% and 13.574% and a STATS accuracy score of 44.58 and 100, respectively. This not only demonstrates the importance of temporal context for accurate road traffic flow prediction but also shows that the temporal context must be relevant. Therefore, using the proposed dynamic temporal context layer has enabled the framework to provide only relevant temporal data segments to the regression model (deep gated recurrent unit model) dynamically in real-time. This led to a 10.8% RMSE and a 40.42 STATS accuracy score improvement in prediction. However, again the accuracy has been a trade-off with the training time taken, with a STATS time score of 14.28 and 0, respectively.

5.5 Summary

In this chapter, we investigated different magnitudes of temporal patterns (short and long-term) by using different temporal data segments to assess how contextual temporal data affects prediction accuracy. Furthermore, we proposed a dynamic temporal context framework. The framework can dynamically incorporate both short and relevant long-term temporal patterns. By using different temporal data segments as input features the model can dynamically determine what temporal segments are needed for regression in real-time. The different temporal data segments and proposed framework were evaluated using an existing real dataset and compared against a comparable prediction model (a deep gated recurrent unit model). The experimental results show that the inclusion of any, short or long-term, temporal pattern improves prediction accuracy. Furthermore, the proposed framework improved prediction accuracy when compared to the deep gated recurrent unit model, with an RMSE of 12.244% and 13.574%, and a STATS accuracy score of 44.58 and 100, respectively.

Based on these experimental results, the temporal patterns within road traffic flow data are vital for accurate prediction. Therefore, could adding more temporal patterns from a road network improve prediction accuracy further? The proposed framework was designed for, and implemented, at one geographical location (RTSS). Extending the model to include road network data would be computationally very heavy. Furthermore, which geographical and temporal input features should be included? There is currently no standard analytical method for the selection of input features on a road network. This issue is complicated further by the very nature of road traffic flow; road traffic flow on a road network is highly interdependent. These interdependent relationships may improve prediction accuracy by providing contextual information for the regression model. However, many input features may be redundant due to the replication of similar temporal patterns (correlation). Therefore, selecting the most discriminative input features for the prediction of road traffic flow remains a challenging task that should be examined further.

Chapter 6

A Novel Dynamic Exogenous Feature Filter using Local Windows for Deep Learning-based Prediction of Road Traffic Flow

Chapter Summary

Due to dynamic input features that change over time and space and are highly correlated, how to select the most discriminative input features remains a challenging task for the accurate prediction of road traffic flow. Therefore, in this Chapter, we have examined the dynamic nature of the spatio-temporal input features' correlations in Section 6.4.2, and developed a novel dynamic exogenous feature selection mechanism based on local windows and Spearman's Rank correlation, in Section 6.3. The proposed method was compared to a state-of-the-art method, a dynamic rolling window feature filter model in Section 6.4.3.

6.1 Introduction

Predicting real-time road traffic flow is a problematic task due to its dynamic and heterogeneous in time and space with its numerous interweaving variables [68] [123] [124], therefore, many researchers only consider one input feature, such as total road traffic flow, for their prediction models [44]. However, a prediction model can only be as good as its input data [40], so including more relevant input features can help improve prediction accuracy as shown by Koesdwiady, Soua, and Karray who added weather conditions [68], and Wang, Zhao, Shao, *et al.* who added truck flows [125]. Yet, in the era of big data, one of the main challenges is what input features should be included in a prediction model. Adding too few input features can cause a model to become too dependent on particular input features, leading to overfitting and poor generalisation. Too many input features and a model may make decisions based on noise, leading to slower training and issues with converging. The issue is exasperated further by adding a spatial dimension to the input data. Although machine learning models such as RNN, LSTM, and GRU have produced promising results [46] [109] [38] extracting temporal features alone is not sufficient.

Spatial dependencies within the traffic data are vital for road traffic prediction [126]. Consequently, spatial-temporal approaches to road traffic flow prediction, that can model the interlacing relationships of the input features both temporally and geographically, have increased in popularity in the last decade. However, what geographical locations should be included? The success of any road traffic prediction model relies heavily on detecting and including these spatial dependencies [127]. Deciding what input features to use in a prediction model is a vital but difficult task with no standard analytical approach.

Current research into feature selection methods can be divided into three categories: 1) wrapper methods, where the usefulness of individual features is based on the prediction model's performance, 2) embedded methods, similar to wrapper methods the usefulness of individual features is based on performance, however, the method is embedded into the prediction model, and 3) filter methods, where useful input features are selected through analysis of their intrinsic properties.

Wrapper methods, such as permutation feature importance [128] and genetic algorithms [129], and embedded methods, such as LASSO [130] and sparse autoencoders [99], despite their popularity, both suffer from a fundamental flaw. They both assume that the relationship between the input features is temporally static, therefore, they only determine the global optimal subset of input features. Road traffic networks are dynamic and suffer from *concept drift*. Concept drift occurs when the probability distributions of the input features change over time. Concept drift can develop suddenly in seconds or slowly evolve over months. The causes of concept drift are, but are not limited to, road traffic incidents, road closures, public holidays, special events, investment/redevelopment of geographical areas, or simply due to the time of day. This can result in input features that were originally deemed vital for road traffic prediction becoming unnecessary, or worse, input features that were regarded as redundant and omitted may now be essential, leading to an ineffective model with poor prediction accuracy. Statically trained models suffer more with concept drift as they can not update the model's weights to account for changes within the road network, however, online trained models [46] are not immune from concept drift. Online models typically update the model's weights when new input data is available, but the input features used are not updated. This can result in redundant input features being used, or worse, vital input features being omitted.

Filter feature selection methods, which included both exogenous and endogenous methods, are computationally very fast. Therefore, despite previous research focusing on the optimal offline global subset of input features, they would lend themselves well to an online setting. Exogenous filters methods use pre-defined spatial 'windows' such as neighbouring nodes or upstream neighbour [131] [132] [133] to determine which input features should be used. However, this method also suffers from a fundamental flaw, it assumes that a road network's dependencies are spatially static and determined only by a node's geographical proximity. Spatial-temporal dependencies in a road network are not limited to adjoining roads or even directly connected roads; relationships between remote nodes can occur due to the interdependent nature of road traffic flows [134]. Furthermore, these relationships may not be a constant. Another issue with exogenous feature filters is that neighbouring nodes often display similar temporal patterns due to road traffic flow's strong periodic patterns. Adding multiple similar input features to a prediction model can cause a model to become too reliant on one temporal pattern, leading to a lack of generalisation. Therefore, a dynamic feature filter not based on spatial proximity is required.

Exogenous feature filters are not dependent on a road network's spatial design and instead filter input features based on statistical analysis. A common endogenous feature filter used for road traffic flow prediction is correlation coefficient (CC) [135]. Most research models that use CC use Pearson's bivariate correlation (PBC) [136] [124] [132]. PBC measures the linear relationship between two sets of data where the variables move in the same relative direction at a constant rate. A more suitable CC analysis would be the Spearman's rank correlation coefficient (SRCC) [137]; SRCC can measure the *monotonic* relationship between two sets of data, therefore, can detect relationships where the variables move in the same relative direction, but not necessarily at a constant rate, detecting more latent relationships within the road traffic data. However, the limited papers that do use SRCC assume that the correlation relationships within the data are fixed in both time and space, and can be adequately described by a global correlation analysis [124]. The global correlation is the measure of the spatial and temporal relationship between two road traffic sensor sites (RTSS) for the whole dataset. However, it is well established that road traffic flow is dynamic and patterns change during the course of a day, week, month and even year [46] [124]. Therefore, a global correlation can not capture all the complex relationships with a road traffic network. The local correlation is a measure of the spatial and temporal relationship between two RTSSs using a subset of the dataset. Some RTSSs may contain strong local correlates yet display no correlation over a larger timespan or geographical area. Therefore, a local correlation would be more suitable.

In this chapter, we propose a framework for road traffic prediction, which contains a novel dynamic exogenous feature filter mechanism using SRCC and *local* windows. This novel feature filter updates the input features delivered to the prediction model to improve prediction accuracy. The proposed framework was compared against three other prediction models: 1) a model with no feature selection, 2) a model with static (global) feature selection, and 3) a state-of-the-art feature selection model using a 'rolling window' [138], which, to the best of our knowledge, has not been applied to road traffic data before.

The contributions and novelty of this chapter include:

1. we investigate the dynamic nature of the road traffic flow's input features by examining the spatial and temporal relationships between RTSSs. We use SRCC to calculate the global and local correlations to determine if a global correlation analysis is adequate enough to illustrate the relationships between the RTSSs.

2. we propose a novel dynamic exogenous feature filter using Spearman's Rank Correlation Coefficient and local windows used in combination with a deep neural network for the prediction of road traffic flow on a network. We compare our proposed model to a deep neural network with no feature selection

filter, a statically (global) chosen feature selection, and a state-of-the-art online *rolling* window [138] feature selection.

6.2 Literature Review of Feature Selection Methods

Feature selection methods (FSMs) are a technique used to select the most important input features for a prediction model. What is defined as important depends upon the method employed. The FSMs can be broadly split into three main categories: 1) wrappers methods (supervised), 2) embedded methods (supervised), and 3) filters methods (unsupervised) [139], as shown in 6.1.

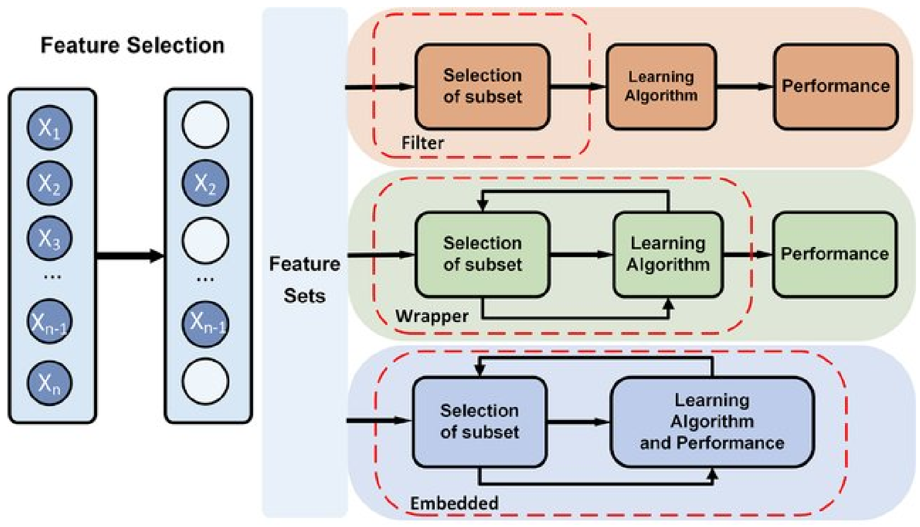

FIGURE 6.1: Feature selection methods, such as filter, wrapper, and embedded method [140]

6.2.1 Wrapper Methods

Wrapper methods measure the usefulness of individual features based on the prediction model's performance. The method performs multiple evaluations of the prediction model using every possible combination of input features to determine the optimal subset of input features, evaluated on an empirical measure such as RMSE or MAPE. Wrappers methods include, but not limited to, permutation feature importance [141], genetic algorithms (GA) [142], and recursive feature elimination [143].

Ou, Xia, Wu, *et al.* [128] used permutation feature importance, derived from random forests, to quantify the importance of each input feature. A 'data-driven feature selection strategy' was then used to select input features for the prediction model (also a random forest). The accuracy of the model, measured by the RMSE, MA, and MAPE, was then compared to 10 other models. The results show that the proposed model was more accurate but no real discussion of the model's complexity or computational time was made. A notable feature of random forests is their ability to measure feature importance, through permutation importance measure, during training without the need for additional training time. However, adding a data-driven FS strategy to the model will add complexity and therefore, computational time to the model. For example, when the proposed model is compared with the ARIMA model the average RMSE is 9.70 and 10.89 respectively. The training time, in seconds, is 350.01 and 0.71. This is a monumental difference which questions the real-life application of the model. Furthermore, the model suffers from a major theoretical flaw. It has assumed that the important input features are static. Past research has shown that road traffic flow is complex and stochastic in nature [68] and therefore, the inter-dependent nature of the input features can change over time.

Chen, Wei, Liu, *et al.* [129] used a GA for feature selection in a proposed novel sparse hybrid GA model for the prediction of road traffic flow. The model contains a sparsity constraint and real encoding scheme in the GA for spatial and temporal input feature selection, reducing the number of input features used in the prediction model. A Least Squares Support Vector Regression (LSSVR) model was then used for prediction. Results show that the proposed model performed better with fewer spatial-temporal variables when compared to other models. However, again no discussion of the model's complexity was had, nor was the training time reported. It would be interesting to compare the prediction models based on model complexity/training time as well as accuracy for a more complete evaluation: e.g. is the trade-off between accuracy and time appropriate or even required? Furthermore, this method also assumes that the relevance of the input features remains static over time.

In conclusion, although wrapper methods can produce accurate results they are prone to overfitting and computationally very heavy [144]. Consequently, they are used to generate a subset of input features that are a constant and not suitable for online dynamic input feature selection.

6.2.2 Embedded Methods

Embedded feature selection methods are similar to wrapper methods as they are also based on the empirical performance of the model. However, the method is embedded into the prediction model, unlike wrapper methods, making them less computationally heavy. Embedded methods included, but not limited to, LASSO method [130], LSTM method [106], and structural risk minimisation method.

Zhou, Hong, Xing, *et al.* [145] proposed a Two Level Hierarchies with time Lag Lasso (TLHL) model for the prediction of road traffic flow based on the idea of combining causality (through time lags) with regression. The model contained a LASSO regression model, which can select input features by shrinking them towards a central point (like the mean) and therefore, eliminating irrelevant input features by reducing them to zero. The proposed model was compared to a regular LASSO with no causality relationship, a LASSO with a fixed time lag causality, and a Granger LASSO. The results showed that TLHL model performed best. The main issue with the LASSO feature selection method for road traffic flow input features is that it assumes that the relationships between the input features are static and therefore, during the training phase determines the global relevant input features. No consideration is given to the dynamic dependencies within the road traffic flow data.

In conclusion, embedded methods are less prone to overfitting and computationally faster than wrapper methods, however, they are still computationally heavy and not suitable for dynamic online learning feature selection.

6.2.3 Filter Methods

Filter feature selection methods, the main category used for road traffic prediction models, selects input features through analysis of their intrinsic properties and can be separated into two distinct sub-classes: 1) exogenous and 2) endogenous.

Exogenous filters, the most commonly used FSM for road traffic flow prediction, are filters based on spatial connectivity within the road structure/network. Popular methods include, but not limited to, only using direct neighbouring nodes, upstream neighbouring nodes, a fixed 'window' of nodes [131] [133], or a network weight matrix [132].

Cao, Ren, and Li [133] transformed their data into a 2-channel 'image like' structure to capture all the nodes' input features. A residual convolutional unit was then applied to model the spatial-temporal relationships of the road traffic flow. Its output was then fused with the output of two other residual neural networks and weights were assigned to different nodes. Cao, Ren, and Li reported that the model's predictions outperformed traditional neural networks with no feature filter. However, while the paper reported good results, the local 'neighbours' in a CNN structure are determined by the window size used. Therefore, assumptions about the road network must be made. This causes two main issues, which can not be solved simply by increasing or decreasing the dimensions of the window. First, a CNN will assume relationships between nodes within the window where no relationship may exist. Similarly, it fails to include relationships between two nodes where a relationship may be present, due to the nodes falling outside the window's boundary. This problem is exasperated further by the second issue, the operation used by CNNs. A CNN uses a co-variance operation that operates a bi-directional relationships between the input features. Consequently, CNNs assume a bi-directional relationship across all its input features within its window. In many real-life data structures, such as road traffic networks, this is not true. A road traffic network structure may include both bi-directional and unidirectional relationships. Furthermore, a bi-directional relationship may only be relevant during saturation periods, such as rush hour periods or during road traffic incidents, resulting in the model searching for irrelevant relationships and adding unnecessary computational time. Therefore, the main issue with an exogenous filter, like the one detailed above, is its rigidity in using pre-defined spatial 'windows'. The filter needs to be more dynamic to truly capture the heterogeneous nature of road traffic flow. Furthermore, its assumptions that the road network's dependencies are static and determined by a node's geographical location can cause issues. While some neighbouring nodes may influence a node's traffic flow, this may not be a constant and there may be relationships between indirectly connected nodes. Therefore, exogenous filters are not suitable for dynamic feature selection.

Endogenous filters are still based on the intrinsic properties (through statistical analysis) of the input features however, they make no assumptions of the road network's spatial design. Endogenous filters include correlation coefficient [135], variance thresholds, and information gains.

Li, Jiang, Li, *et al.* [132] noted that adding all input features to a prediction model was not the best practice and sought to compare feature selection methods for road traffic flow prediction models. Li, Jiang, Li, *et al.* compared three different methods: 1) graphical LASSO, 2) geographical neighbours (a network weight matrix), and 3) correlations. The results were mixed, showing no clear superior method. However, the research used Pearson's bivariate correlation (PBC) [136]. PBC is a measure of the linear relationship between two sets of data, where the variables move in the same relative direction at a constant rate. Using a different correlation calculation, such as SRCC [137] may have produced different results. SRCC can measure the monotonic relationship between two sets of data, therefore, can detect relationships where the variables move in the same relative direction, but not necessarily at a constant rate. Furthermore, Li, Jiang, Li, *et al.* assumed that the correlation relationships within the data were fixed in both time and space and therefore, only calculated a global (static) correlation coefficient.

In conclusion, filter methods are computationally less heavy when compared with wrapper and embedded methods, therefore, endogenous filters, in particular, lend themselves well to online learning.

6.3 Methodology

We have proposed a framework containing an online prepossessing component in the input layer to select relevant input features. The novel dynamic exogenous feature filter uses Spearmen's rank correlation coefficient and three predefined temporal windows, known as local windows, to periodically update the input features in real-time before it is passed through to the regression model. The model is defined in more detail below. Furthermore, for the regression model used within the ensemble framework architecture, we have extended our online model [46] to allow for multiple RTSSs (a road network) to be added to the input data. All adaptations are detailed below.

The framework is divided into three components: 1) the input layer, 2) the model layer, and 3) the update scheme layer, as seen in Fig. 6.2.

6.3.1 Input Layer

The input layer comprises the input data, including the current network data segments and the historical network data segments, and the dynamic exogenous feature filter mechanism. Each element will be defined in more detail.

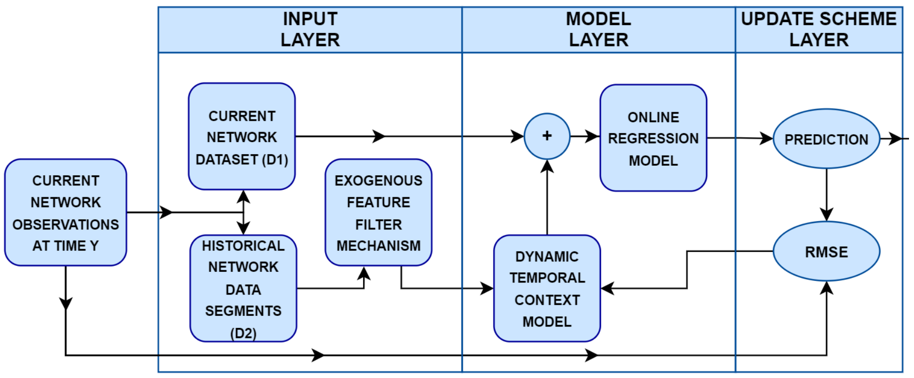
INPUT LAYER
MODEL LAYER
UPDATE SCHEME LAYER
CURRENT NETWORK DATASET (D1)
CURRENT NETWORK OBSERVATIONS AT TIME Y
HISTORICAL NETWORK DATA SEGMENTS (D2)
EXOGENOUS FEATURE FILTER MECHANISM
+
ONLINE REGRESSION MODEL
DYNAMIC TEMPORAL CONTEXT MODEL
PREDICTION
RMSE

FIGURE 6.2: The proposed framework

Input Data

The proposed ensemble framework has two sources of input data from the road network for the input data layer. The two sources of data are: 1) the current network observations dataset (D_1), and 2) the historical temporal and spatial data segments (D_2).

The current network observations dataset (D_1) is a 3D array. D_1 contains six input features (f). The input features are the total traffic flow categorised into six different vehicle types ($f_i \Rightarrow i \in \mathbb{Z} : 1 \leq i \leq 6$), as shown in Table 3.1, overtime (t) from all 13 different RTSSs ($s_n \Rightarrow n \in \mathbb{Z} : 1 \leq i \leq 13$) as shown in Table 6.1. Vehicle classes were used as input features (f) for the prediction model based on previous research [121] that demonstrated that vehicle classes have their own latent patterns and including them as different input features improves prediction accuracy.

The historical temporal and spatial data segments dataset (D_2) is a 3D array, as shown in Figure 6.3. D_2 contains the historical temporal data segments for all 13 RTSSs (S). The dataset's time steps have a time horizon of 5 minute, therefore, the historical temporal data segments are historical data from the previous day ($d = t - 288$), week ($w = t - 2,016$), and month ($m = t - 8,064$) prior to the point of prediction ($t + 1$). Due to COVID-19 affecting the input data a one-year historical temporal segment was omitted. Each temporal data segment contains the breakdown of the total traffic flow into six different vehicle types (as shown in Table 3.1, mirroring the current network dataset, D_1. Both the current network dataset (D_1) and the historical temporal and spatial data segments (D_2) are passed to the online correlation filter, while the current network dataset (D_1) is also fed directly to the regression model (gated recurrent model detailed below).

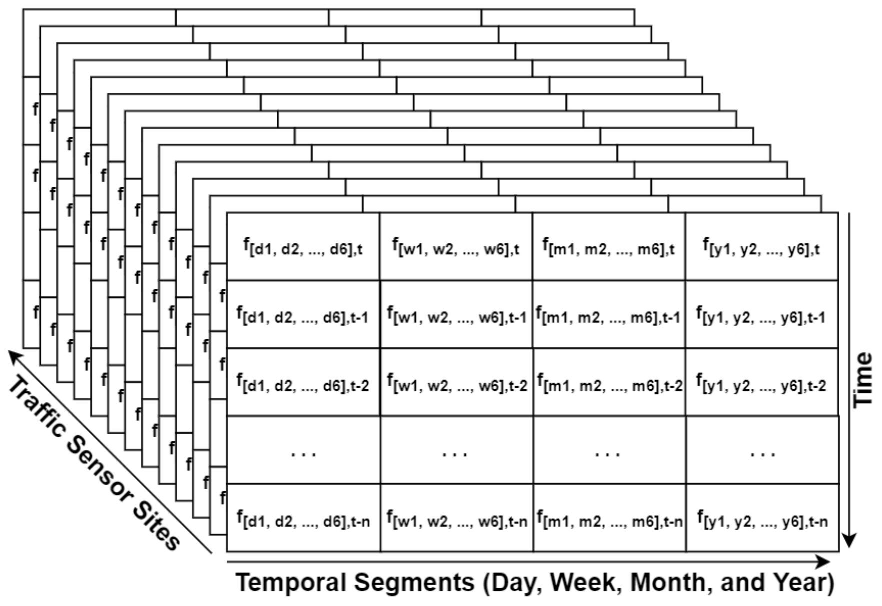
$f_{[d1, d2, ..., d6],t}$
$f_{[w1, w2, ..., w6],t}$
$f_{[m1, m2, ..., m6],t}$
$f_{[y1, y2, ..., y6],t}$
$f_{[d1, d2, ..., d6],t-1}$
$f_{[w1, w2, ..., w6],t-1}$
$f_{[m1, m2, ..., m6],t-1}$
$f_{[y1, y2, ..., y6],t-1}$
$f_{[d1, d2, ..., d6],t-2}$
$f_{[w1, w2, ..., w6],t-2}$
$f_{[m1, m2, ..., m6],t-2}$
$f_{[y1, y2, ..., y6],t-2}$
$f_{[d1, d2, ..., d6],t-n}$
$f_{[w1, w2, ..., w6],t-n}$
$f_{[m1, m2, ..., m6],t-n}$
$f_{[y1, y2, ..., y6],t-n}$
Traffic Sensor Sites
Time
Temporal Segments (Day, Week, Month, and Year)

FIGURE 6.3: The historical network data (D_2) which includes temporal segments of a day, week, month and year

Dynamic Exogenous Feature Filter Mechanism

The novel dynamic exogenous feature filter mechanism aims to remove unnecessary or redundant input features before they are passed through to the model layer. In the era of big data, vast amounts of input features are available to be incorporated into regression models. While additional input features, such as different vehicle classes, can improve prediction accuracy [46] adding unnecessary or redundant input features can negatively affect the computational time of the model and increase the likelihood of overfitting. Therefore, a vital part of model development is filtering out unnecessary and redundant input features.

Choosing an appropriate data filter is entirely dependent on the data type and its characteristics. Road traffic flow on a road network, due to its strong periodic patterns, is highly correlated. This correlation can lead to some input features becoming redundant. Therefore, a correlation filter would be suitable for road traffic flow data.

To calculate the crosscorrelation the Spearman's rank correlation coefficient (SRCC) [137] was used. Most research papers on correlation and road traffic use Pearson's bivariate correlation (PBC) [136] [124], however, SRCC was chosen over PBC due to SRCC being non-parametric. SRCC can measure the *monotonic* relationship between two sets of data, therefore, can detect relationships where the variables move in the same relative direction, but not necessarily at a constant rate. PBC is a measure of the linear relationship between two sets of data, therefore, can only detect linear relationships where the variables move in the same relative direction at a constant rate. SRCC is also less sensitive to outliers. Road traffic flow, especially in urbanised areas, can be volatile and susceptible to sudden changes due to road traffic incidents or road traffic flow reaching a road's saturation point. Therefore, the model must be able to cope with outliers. Lastly, PBC assumes the input data is homoscedastic; road traffic flow is heteroscedastic.

SRCC, used for road traffic flow correlation where the RTSS one (s^1) is the point of prediction can be defined as

$$s_r = \rho R(s^1_{f_i}), R(s^m_{f_i}) = \frac{COV(R(s^1_{f_i}), R(s^m_{f_i}))}{\sigma_{R(s^1_{f_i})}, \sigma_{R(s^m_{f_i})}} \tag{6.1}$$

where ρ denotes the Pearson's correlation, f_i is the input feature ($f_i \Rightarrow i \in \mathbb{Z} : 1 \leq i \geq 6$), and s^n denotes the RTSS. In this equation $s^m \Rightarrow m \in \mathbb{Z} : 2 \leq m \geq 13$. RTSS one is excluded as it is the point of prediction; the site all other sites are compared to. Therefore, RTSS one (s^1) is included on the left side of Equation 6.1.

Therefore, if all points in the input feature's dataset $(F_i = f_{i,t}, f_{i,t-1}, f_{i,t-n})$ ranks are all distinct integers then

$$s_r = 1 - \frac{6\sum d^2_{(f_i,t_n)}}{n(n^2-1)} \tag{6.2}$$

where

$$d_{(f_i,t_n)} = R\left(s_{1_{(f_i,t_n)}}\right) - R\left(s_{n_{(f_i,t_n)}}\right) \tag{6.3}$$

One issue with SRCC, and PBC, is that it computes the global relationship (correlation) between the input features within the dataset. However, as previously stated, road traffic flow is complex and stochastic [68]; it is both dynamic and heterogeneous in time and space [124]. Therefore, the correlation relationships within the data will be dynamic, and any model used to assess the correlation in road traffic flow should accommodate this dynamic nature.

We propose a dynamic *local windows* mechanism. The dynamic local windows mechanism uses three predefined temporal windows (w). For the input features (f_i and $f_i \Rightarrow i \in \mathbb{Z} : 1 \leq i \geq 6$) as stated in Table 3.1, *cut points* (k_1, k_2), are, in ascending order, selected for f_1 and f_i. It is generally accepted that road traffic patterns in urbanised areas during weekdays display three distinct sections: 1) morning rush hour peak, 2) midday trough, and 3) evening rush hour peak [124], which can be seen in Figure 6.5. Therefore, these smaller patterns (7:00AM to 11:00AM, 11:00AM to 3:00PM, and 3:00PM to 7:00PM) will be used for the local windows for weekday road traffic flow data. A two-dimensional matrix, denoted as K is then produced to store the data points that fall within the cut points (k_1, k_2). The only data that will be used to calculate the local correlations within the windows will be the training data, as defined in the Experimental Evaluation section.

Dynamic Rolling Window Filter Feature

We also present a second model for comparison, a state-of-the-art model which also uses local correlations. The model calculates the local correlations through an online rolling window algorithm [138] for streaming data. Therefore, we have adapted this methodology for road traffic flow. The online dynamic model which uses streaming data and a *rolling window* is used during the testing process to continually update the K matrix. The model uses the previous static windows model's input feature selection as a starting point then as a new data point $\left(s^1(f_{1,t+1}, ..., f_{6,t+1}), ..., s^{13}(f_{1,t+1}, ..., f_{6,t+1})\right)$ becomes available the cut points are moved along from k_1 and k_2 to $k_1 + 1$ and $k_2 + 1$ and the new data point is added to the K matrix while the oldest data point is removed. The SRCC is then recalculated at every model iteration. The size of the window will be determined during the testing phase using brute force trial and error.

Therefore, given the site of prediction, RTSS one $(s^1 \in f_{1,t_n}, f_{1,t_2}, ..., f_{6,t_n})$, the SRCC can be calculated as

$$s_r \frac{\sum_{q=1}^{t}\left(R(s^1_{f_i})_q - \overline{R(s^1_{f_i})}\right)\left(R(s^i_{f_i})_q - \overline{R(s^i_{f_i})}\right)}{\sqrt{\sum_{q=1}^{t}\left(R(s^1_{f_i})_q - \overline{R(s^1_{f_i})}\right)^2 \sum_{q=1}^{t}\left(R(s^i_{f_i})_q - \overline{R(s^i_{f_i})}\right)^2}} \tag{6.4}$$

The current traffic flow data at RTSS one is fed directly to the regression layer, while the remaining traffic flow data (various sites and temporal segments) is updated based on the SCRR results and any highly correlated input features are removed, then passed through to the regression layer.

6.3.2 The Model and Update Scheme Layer

Building on our previous work in Chapter 5 and paper [46], we have extended the model layer to allow for for multiple RTSSs in the input data. The model layer includes two different neural network architectures: 1) a dynamic temporal context (DTC) model architecture, and 2) a regression model architecture.

Different to existing time-series models using CNN, where the prediction models are based on static data, the DTC model is dynamic and seeks to find a relationship between different magnitudes of temporal data segments. The DTC model takes in various magnitudes of temporal data, D_2 (one hour, one day, one week, one month, one year), and uses a convolutional neural network (CNN) [35] to create the current dataset (C), which is then passed on to the regression model. The model's architecture has been adapted to include more nodes to accept multiple RTSS data, such as dataset D_2, as shown in Figure 6.3.

The keys points of the DTC model are;

1. Temporal segments are used as an input feature, therefore, the CNN's kernel (k) scrolls across the temporal data and not down like in traditional CNN architectures.

2. The kernel (k) used to scroll across the temporal data segments (D_2) is rectangular and not square as in traditional CNN arrangements. Instead, a matrix the size of one by the number of input features within the temporal data segment is used. Therefore, it only considers (convolves) all the data within the temporal data segment at one time point at once and at one RTSS.

3. The kernel's (k) stride in typical CNN setups is one. However, in the DTC model, the stride is the same size number of input features within the temporal data segment. Therefore, each temporal data segment is only considered once every pass.

4. The model does not use any 'padding' which is typically used in CNN architectures to return the input data to its original size. Downsampling is purposely used so the model only retains the most relevant temporal features.

The current dataset, C, is passed through to the regression architecture. For the regression, based on previous research [122], the model uses a deep a gated recurrent unit (GRU) architecture. A GRU model works through the use of gates; each gate is a neural network. The gates included in a standard GRU cell are an *update gate* and a *forget gate*, as shown in Figure 4.3 to make predictions.

A more detailed explanation of the DTC model can be found in Section 5.3.1 and the published paper [46].

6.4 Experimental Evaluation

In this section we focus on two research questions: 1) is a global correlation sufficient to capture the relationships between the road traffic flows' spatial and temporal input features? 2) can a novel dynamic exogenous feature filter method for the prediction of road traffic flow on a network improve prediction accuracy?

6.4.1 Data Description

A real dataset was used for experimentation. The dataset contained thirteen RTSSs around Manchester city centre, each site is positioned on all the primary roads that join the inner ring road encircling Manchester's city centre (as seen in Figure 6.4).

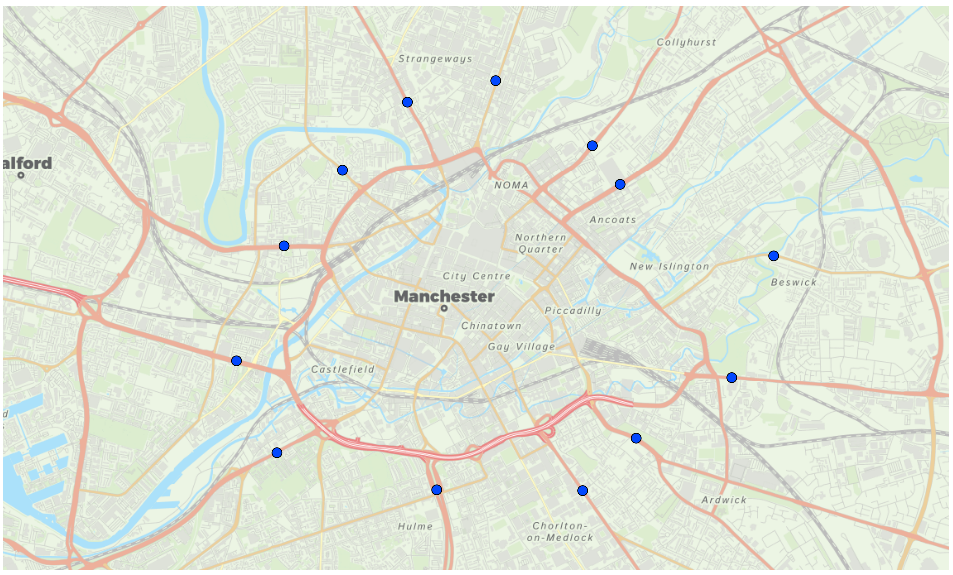

FIGURE 6.4: A map of Manchester illustrating the geographical locations of the thirteen RTSSs, depicted as blue dots, chosen for the test case

A detailed list of the RTSSs and their orientation can be found in Table 6.1. The dataset contained data from 1st January 2020 to 31st March 2020 (90 days) and had a time step of five minutes (25920 time points). Each RTSS also contained input features of different vehicle classes, as shown in Table 3.1, as different vehicle classes have been shown to improve prediction accuracy [121]. Therefore, the input data is a 3D array of 7 by 25920 by 13.

Site Name	Geographical Location	Orientation
1026	Chapel St (A6) / 5m W of Great George St, Salford (ATC)	E
1161	Princess Rd (A5103) / 100m N of Bonsall St, Hulme, Manchester (ATC)	N
1165	Ardwick Green (A6) / 35m E of Hamsell Rd, Ardwick, Manchester (ATC)	SE
1368	Cheetham Hill Rd (A665) / 15m N of Knowsley St, Collyhurst, Manchester (ATC)	N
1410	Regent Rd (A57) / 105m W of Ordsall Ln, Salford (ATC)	E
1412	Oldham Rd (A62) / 40m SW of Poland St, Ancoats, Manchester (ATC)	NE
1413	Rochdale Rd (A664) / 20m SW of Peary St, Ancoats, Manchester (ATC)	SW
1414	Upper Brook St (A34) / 80m N of Cottenham St, Manchester (ATC)	SE
1415	Chester Rd (A56) / 25m SW of Barrack St, Manchester (ATC)	NE
1416	Ashton Old Rd (A635) / 90m E of Chancellor Ln, Ancoats, Manchester (ATC)	E
1418	Ashton New Rd (A662) / 130m W of Hillkirk St, Manchester (ATC)	E
1419	Great Ducie St (A56) / 10m S of Sherborne St, Manchester (ATC)	SE
1420	Blackfriars Rd (A6041) / 15m NW of Greengate W, Salford (ATC)	SE

TABLE 6.1: Details of the RTSS's geographical location, including their orientation.

6.4.2 Research Question One

Road traffic flow is dynamic, its temporal, spatial, and heterogeneous patterns can change daily, weekly, monthly, and even yearly [46]. Furthermore, these patterns evolve, due to concept drift, or can suddenly shift due to road traffic incidents. Therefore, the relationships between road traffic flows at different geographical locations (RTSSs) are also dynamic. In this section, we aim to answer research question one, is a global correlation sufficient to capture the relationships between the road traffic flows' spatial and temporal input features, by investigating the dynamic nature of these input features (road traffic flow broken down by vehicle class). We examine their dynamic relationships by analysing their global and local correlations.

The total traffic flow training data for each RTSSs location (22 days from 7:00AM to 7:00PM) was used to create an 'average day', as shown in Figure 6.5. Bank holidays and weekends were omitted due to road traffic patterns shifting considerably from weekday to weekend.

Each RTSSs location shows a similar daily pattern, with differences in volume being attributed to the different roads' capacities. This was to be expected as it is well established that road traffic flow in urbanised areas exhibits strong daily periodic patterns (autocorrelated) [124]. However, despite using the averages the plotted lines are still extremely jerky and not smooth; this indicates the road traffic flow's dynamic and heterogeneous nature.

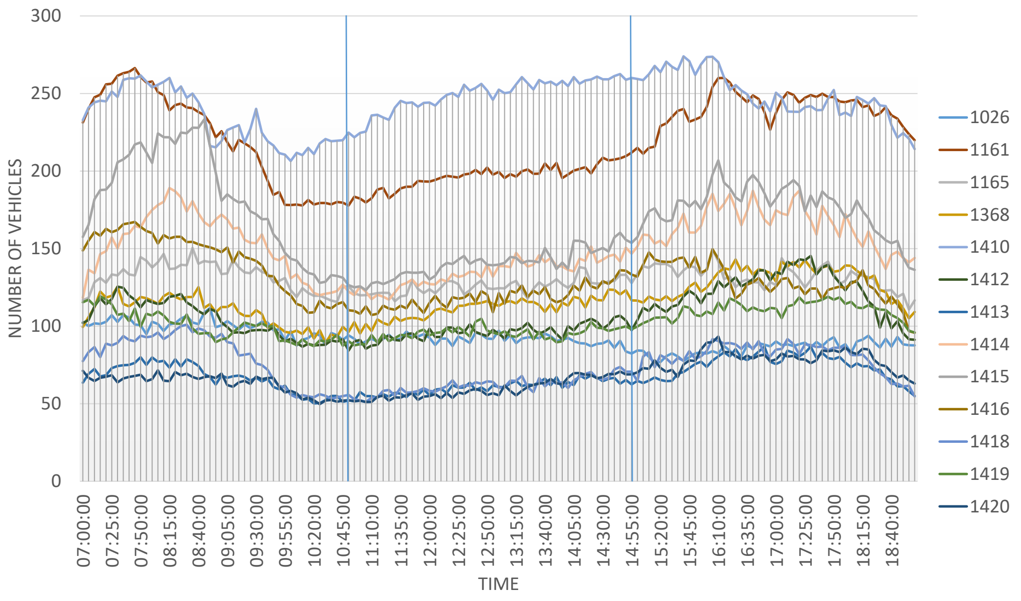

FIGURE 6.5: The average weekday road traffic flow for all thirteen RTSSs

It is generally accepted that traffic patterns in urbanised areas display three distinct periods during weekdays: 1) morning rush hour peak, 2) midday trough, and 3) evening rush hour peak [124], which can be seen in Figure 6.5. Therefore, these smaller patterns will be investigated further when exploring the input feature's correlation changes over time. These three distinct periods will be the local correlations.

First, the crosscorrelation of all the weekdays (excluding bank holidays), across all sites, using all vehicle classes (Table 3.1), was calculated (Equation 6.4) the point of prediction (RTSS 1161, as shown in Table 6.1) as the reference point. This is referred to as the *global correlation* hereafter. Next, the correlations for each of the three temporal segments (7:00AM to 11:00AM, 11:00AM to 3:00PM, and 3:00PM to 7:00PM) shown in Figure 6.5, known as the local correlations, were calculated and the results were compared.

Table 6.2 show that the global correlation input feature two, cars or vans, is highly correlated across all sites, with an average crosscorrelation of 90%. Geographical distance from the reference site (1161) bears no influence on the correlation. This is most likely due to input feature two, cars or vans, being the dominant input feature and with strong periodic patterns due to commuting traffic. Variable four, rigid goods, is also well correlated with an average correlation of 48%, however, it has a large range (26% to 5%) when compared to input feature two (82% to 92%). The

		7AM to 7PM						7AM to 11AM						11AM to 3PM						3PM to 7PM					
		1161						1161						1161						1161					
		1	2	3	4	5	6	1	2	3	4	5	6	1	2	3	4	5	6	1	2	3	4	5	6
1026	1	0.27	0.60	0.15	0.53	0.20	0.08	0.04	0.50	0.15	0.47	0.07	-0.06	0.03	0.08	0.01	0.01	-0.01	-0.04	0.06	0.31	0.01	0.09	-0.01	-0.02
	2	0.37	0.82	0.20	0.79	0.23	0.11	0.06	0.47	0.11	0.41	0.14	0.03	0.06	0.11	-0.01	-0.06	0.10	-0.05	0.00	0.18	0.01	0.04	0.02	-0.06
	3	0.10	0.18	0.03	0.13	0.06	0.01	0.04	0.08	-0.05	0.00	0.07	0.06	0.02	0.04	-0.01	0.00	0.04	0.01	0.05	0.06	0.00	0.06	-0.03	-0.03
	4	0.38	0.66	0.16	0.58	0.35	0.10	0.14	0.06	-0.01	-0.01	0.07	0.10	0.12	0.13	0.02	-0.10	0.16	-0.03	0.02	0.13	0.02	0.05	0.10	0.02
	5	0.31	0.62	0.15	0.52	0.23	0.08	0.08	0.23	0.09	0.20	0.04	0.08	0.06	0.18	0.04	-0.05	0.08	0.04	0.02	0.12	0.00	-0.03	0.06	0.01
	6	0.35	0.65	0.15	0.58	0.29	0.10	0.07	0.05	0.01	0.08	0.07	0.09	0.09	0.15	0.05	-0.07	0.14	-0.03	0.02	0.14	0.01	-0.03	0.10	0.04
1165	1	0.25	0.46	0.11	0.41	0.21	0.08	0.08	0.16	0.06	0.16	-0.02	-0.01	0.09	0.08	-0.01	0.02	0.12	-0.01	0.06	0.09	0.04	-0.01	0.11	0.05
	2	0.41	0.91	0.22	0.82	0.26	0.12	0.09	0.64	0.14	0.60	0.10	0.00	0.09	0.37	-0.01	-0.15	0.01	-0.04	0.04	0.27	0.06	0.08	0.13	0.06
	3	0.12	0.22	0.03	0.19	0.13	0.02	0.03	0.10	-0.03	0.08	0.02	-0.03	-0.02	0.05	-0.06	0.00	0.01	-0.05	-0.03	0.03	0.02	0.06	0.06	0.04
	4	0.35	0.56	0.13	0.51	0.40	0.11	0.15	-0.12	0.02	-0.13	0.12	0.17	0.11	0.05	-0.01	-0.05	0.16	0.01	0.01	-0.04	0.00	0.03	0.21	0.13
	5	0.24	0.36	0.08	0.31	0.26	0.09	0.05	-0.01	0.01	-0.03	0.08	0.12	0.16	0.08	-0.01	-0.07	0.11	0.03	0.03	0.02	-0.02	0.02	0.09	0.06
	6	0.38	0.73	0.16	0.64	0.30	0.13	0.12	-0.34	-0.08	-0.33	0.13	0.21	0.07	0.07	-0.01	0.00	0.12	-0.01	0.05	0.14	-0.02	0.00	0.06	0.00
1368	1	0.27	0.55	0.12	0.47	0.17	0.04	0.07	0.06	0.03	0.11	0.10	0.05	0.09	0.21	-0.02	-0.10	0.04	-0.04	0.06	0.15	0.00	0.06	0.05	-0.04
	2	0.41	0.92	0.21	0.80	0.22	0.09	0.07	0.66	0.13	0.57	0.08	-0.03	0.02	0.33	0.00	-0.08	0.05	-0.06	0.04	0.43	0.00	0.07	-0.01	-0.05
	3	0.09	0.12	0.03	0.10	0.09	0.02	-0.05	-0.04	-0.03	-0.04	0.07	0.00	0.09	0.04	0.03	0.00	-0.04	-0.03	0.02	0.02	-0.04	0.03	0.07	0.03
	4	0.33	0.58	0.14	0.51	0.32	0.10	0.09	-0.07	-0.02	-0.11	0.06	0.17	0.09	0.08	-0.04	-0.04	0.08	0.02	-0.01	0.04	-0.02	0.03	0.12	0.07
	5	0.09	0.17	0.05	0.15	0.12	0.05	0.00	0.09	0.03	0.08	0.00	-0.02	0.03	0.05	-0.01	-0.06	0.08	0.05	-0.01	-0.01	-0.01	0.04	0.06	0.03
	6	0.21	0.44	0.11	0.43	0.18	0.10	0.07	0.04	0.02	0.03	0.04	0.05	0.06	0.10	-0.02	-0.02	0.08	-0.05	0.01	0.05	0.01	-0.02	0.04	-0.03
1410	1	0.22	0.46	0.10	0.43	0.13	0.06	0.05	0.28	0.04	0.24	0.05	-0.02	0.05	0.12	-0.05	-0.02	0.04	0.02	0.04	0.15	-0.01	0.03	-0.01	-0.03
	2	0.41	0.90	0.21	0.79	0.26	0.08	0.09	0.66	0.17	0.48	0.08	-0.10	0.10	0.44	0.00	-0.21	0.09	-0.03	-0.04	0.09	0.05	0.11	0.08	0.06
	3	0.18	0.33	0.08	0.29	0.20	0.04	0.05	0.12	0.05	0.05	0.07	0.03	0.04	0.12	0.01	-0.02	0.04	0.02	-0.03	0.00	-0.05	0.12	0.04	0.03
	4	0.31	0.45	0.11	0.40	0.42	0.09	0.15	-0.09	-0.05	-0.13	0.14	0.16	0.09	0.04	-0.03	-0.08	0.20	0.02	0.07	0.01	-0.01	0.04	0.16	0.08
	5	0.28	0.42	0.10	0.37	0.33	0.09	0.13	0.13	0.00	0.13	0.11	0.16	0.09	0.07	0.02	-0.04	0.15	0.04	0.10	0.19	0.01	-0.01	0.04	0.04
	6	0.13	0.27	0.07	0.27	0.12	0.05	0.02	0.10	0.01	0.08	0.00	0.02	-0.02	0.08	0.03	0.01	0.05	-0.06	0.02	0.01	-0.03	-0.05	-0.03	0.06
1412	1	0.19	0.37	0.09	0.34	0.13	0.04	0.05	0.20	-0.01	0.16	0.04	-0.02	0.04	0.06	0.03	-0.01	0.01	-0.01	0.05	0.11	0.01	-0.02	0.06	0.00
	2	0.42	0.92	0.22	0.82	0.24	0.09	0.11	0.68	0.15	0.54	0.09	-0.08	0.05	0.30	0.07	-0.04	-0.01	-0.08	0.06	0.39	0.00	0.08	-0.02	-0.02
	3	0.07	0.16	0.04	0.16	0.09	0.05	0.00	0.04	0.01	0.08	-0.02	0.03	-0.02	0.05	0.02	-0.01	-0.02	-0.02	-0.04	-0.01	-0.02	0.07	0.06	0.01
	4	0.25	0.31	0.08	0.26	0.37	0.08	0.15	-0.03	0.02	0.00	0.10	0.14	0.13	0.08	-0.03	-0.06	0.16	0.01	0.03	-0.01	0.02	0.00	0.18	0.08
	5	0.16	0.24	0.06	0.21	0.18	0.08	0.08	0.05	-0.01	0.07	0.03	0.04	0.00	0.01	0.00	0.04	0.11	0.02	0.04	-0.01	-0.02	0.01	0.07	0.03
	6	0.36	0.69	0.16	0.62	0.29	0.10	0.05	0.03	-0.01	0.15	0.11	0.11	0.07	0.13	0.01	-0.03	0.07	0.01	0.05	0.17	-0.02	0.03	0.11	0.01
1413	1	0.09	0.18	0.05	0.17	0.05	0.03	0.04	0.15	0.02	0.17	-0.04	0.02	0.01	0.07	0.01	0.00	-0.02	0.00	0.04	0.12	0.07	0.02	-0.04	0.00
	2	0.41	0.91	0.21	0.82	0.25	0.09	0.10	0.69	0.18	0.60	0.10	-0.09	0.02	0.29	-0.03	-0.04	0.07	-0.05	0.03	0.34	-0.02	0.11	0.03	-0.03
	3	0.08	0.16	0.03	0.12	0.06	-0.01	0.03	-0.01	0.01	0.01	-0.01	-0.04	-0.02	0.01	-0.03	-0.05	0.07	0.00	0.02	0.11	0.01	0.04	0.00	-0.06
	4	0.28	0.46	0.11	0.40	0.32	0.09	0.09	0.03	0.02	0.02	0.11	0.08	0.12	0.13	-0.06	-0.06	0.17	-0.02	-0.01	0.19	0.02	0.07	0.10	-0.01
	5	0.14	0.28	0.07	0.23	0.11	0.05	0.03	0.01	-0.05	-0.05	0.08	0.03	0.01	0.03	0.01	0.01	-0.01	-0.03	0.00	0.15	0.05	0.04	-0.02	0.03
	6	0.37	0.72	0.17	0.65	0.30	0.10	0.12	0.05	0.00	0.06	0.12	0.10	0.05	0.14	0.05	0.00	0.12	-0.03	0.05	0.19	0.04	0.01	0.08	0.02
1414	1	0.23	0.49	0.11	0.42	0.12	0.10	0.08	0.20	0.07	0.18	0.07	0.06	0.02	0.13	-0.02	-0.05	0.05	0.02	0.01	0.17	0.01	0.03	-0.01	0.00
	2	0.42	0.91	0.22	0.80	0.25	0.13	0.08	0.62	0.14	0.60	0.11	0.03	0.12	0.48	-0.02	-0.25	0.10	-0.04	0.01	0.32	0.06	0.17	0.12	0.01
	3	0.08	0.13	0.02	0.10	0.08	0.00	0.07	0.02	0.01	0.02	0.02	-0.02	0.02	0.07	-0.05	-0.05	0.05	-0.02	0.02	0.00	-0.01	0.00	0.07	-0.03
	4	0.31	0.55	0.14	0.49	0.34	0.11	0.11	0.09	0.01	0.05	0.14	0.09	0.02	0.06	0.05	-0.06	0.14	-0.01	-0.02	0.04	0.00	0.06	0.14	0.14
	5	0.21	0.38	0.08	0.29	0.20	0.05	0.07	0.09	-0.02	0.11	0.11	-0.01	0.07	0.14	0.01	-0.12	0.09	-0.01	0.04	0.12	0.00	0.01	0.11	0.06
	6	0.19	0.36	0.10	0.33	0.15	0.09	0.05	0.07	0.04	0.14	0.09	0.08	0.03	0.07	0.03	-0.03	0.06	-0.01	-0.07	0.09	-0.02	0.03	-0.04	-0.03
1415	1	0.21	0.47	0.11	0.44	0.11	0.06	0.03	0.38	0.04	0.34	0.04	-0.02	0.03	0.11	0.00	-0.01	-0.01	0.00	0.02	0.12	0.01	0.01	0.00	-0.05
	2	0.42	0.92	0.22	0.84	0.26	0.13	0.08	0.74	0.16	0.66	0.06	-0.07	0.12	0.50	0.02	-0.19	0.06	-0.08	0.02	0.38	0.04	0.20	0.04	0.02
	3	0.16	0.37	0.10	0.32	0.11	0.07	0.05	0.27	0.03	0.20	0.05	0.03	-0.02	0.06	-0.01	0.03	0.04	0.02	0.05	0.15	0.03	0.06	0.02	0.04
	4	0.39	0.72	0.17	0.65	0.35	0.13	0.12	0.08	-0.01	0.00	0.11	0.10	0.13	0.16	-0.04	-0.13	0.06	-0.01	-0.03	0.01	-0.01	0.07	0.18	0.13
	5	0.31	0.51	0.13	0.44	0.34	0.09	0.13	0.17	0.05	0.11	0.08	0.09	0.07	0.11	-0.04	-0.07	0.10	-0.01	0.00	0.11	0.03	0.04	0.16	0.09
	6	0.10	0.17	0.05	0.16	0.07	0.07	0.02	-0.02	-0.01	-0.05	-0.02	0.10	0.03	0.05	0.03	-0.02	0.10	0.00	0.06	0.02	0.01	-0.08	-0.02	-0.03
1416	1	0.29	0.64	0.14	0.58	0.19	0.07	0.04	0.22	0.03	0.17	0.03	-0.02	0.06	0.06	-0.02	0.00	0.07	0.03	0.01	0.20	0.03	0.13	0.04	0.03
	2	0.42	0.88	0.22	0.82	0.32	0.13	0.06	0.76	0.16	0.68	0.11	-0.06	0.10	0.42	0.03	-0.17	0.04	-0.05	0.01	0.12	0.01	0.05	0.11	0.11
	3	0.19	0.45	0.10	0.36	0.11	0.03	0.01	0.17	0.04	0.08	0.05	0.00	0.05	0.14	-0.02	-0.05	0.03	-0.04	-0.05	0.17	-0.01	0.09	0.05	0.03
	4	0.35	0.58	0.15	0.51	0.39	0.10	0.12	-0.06	-0.03	-0.08	0.13	0.12	0.12	0.12	0.00	-0.05	0.15	0.04	0.00	0.06	0.03	0.07	0.14	0.10
	5	0.25	0.38	0.10	0.31	0.30	0.08	0.10	0.12	0.00	0.07	0.06	0.09	0.07	0.09	-0.02	-0.05	0.10	0.00	0.01	0.07	0.05	0.08	0.11	0.04
	6	0.12	0.25	0.06	0.23	0.11	0.03	0.01	0.06	-0.02	0.10	-0.01	0.00	-0.02	0.03	-0.01	-0.01	0.01	0.03	-0.01	0.03	0.03	0.02	0.04	0.03
1418	1	0.07	0.12	0.02	0.11	0.04	0.04	0.04	0.08	0.03	0.10	0.03	0.05	0.07	0.01	-0.01	0.02	0.01	-0.03	0.06	0.05	-0.06	-0.02	0.01	-0.01
	2	0.41	0.90	0.22	0.84	0.28	0.11	0.04	0.69	0.18	0.70	0.08	-0.04	0.03	0.35	0.03	-0.10	-0.01	0.01	-0.01	0.26	0.03	0.14	0.04	-0.01
	3	0.08	0.17	0.06	0.16	0.08	0.03	0.03	0.07	0.02	0.08	0.03	0.03	-0.02	0.07	0.02	-0.06	-0.03	0.02	-0.03	0.01	0.00	0.08	0.03	-0.04
	4	0.30	0.48	0.12	0.42	0.36	0.09	0.09	0.02	-0.03	0.06	0.12	0.15	0.09	0.16	0.02	-0.07	0.11	-0.01	0.03	0.02	0.00	0.08	0.16	0.11
	5	0.14	0.24	0.06	0.19	0.18	0.04	0.03	0.01	0.03	0.00	0.05	0.05	-0.02	0.07	-0.04	-0.04	0.06	0.00	0.05	0.08	0.01	-0.04	0.03	0.01
	6	0.30	0.58	0.14	0.54	0.23	0.07	0.05	0.06	0.04	0.12	0.03	0.05	0.08	0.10	0.00	0.02	0.11	0.01	0.03	0.09	0.00	0.00	0.05	-0.02
1419	1	0.12	0.25	0.07	0.21	0.08	0.03	0.08	0.15	0.05	0.14	0.02	0.01	-0.01	0.01	0.05	-0.01	0.05	-0.07	0.06	0.11	0.00	-0.01	0.00	-0.07
	2	0.41	0.88	0.20	0.79	0.23	0.11	0.08	0.58	0.17	0.48	0.11	-0.03	0.01	0.22	-0.02	0.02	0.03	-0.09	0.03	0.29	0.02	0.04	0.01	0.04
	3	0.16	0.33	0.09	0.26	0.12	0.02	0.05	0.03	0.01	0.02	0.03	0.04	0.03	0.15	0.02	-0.04	0.03	-0.02	-0.04	0.07	0.03	0.07	0.04	0.00
	4	0.35	0.62	0.14	0.52	0.32	0.10	0.13	-0.16	-0.05	-0.18	0.09	0.18	0.08	0.11	0.01	-0.02	0.09	-0.02	0.02	0.11	-0.02	0.08	0.12	0.06
	5	0.24	0.45	0.11	0.37	0.23	0.04	0.05	0.16	0.04	0.15	0.07	0.00	0.11	0.16	0.02	-0.07	0.16	-0.05	-0.03	0.15	0.03	0.08	0.07	0.01
	6	0.30	0.57	0.13	0.50	0.24	0.06	0.07	0.04	0.03	0.07	0.10	0.09	0.06	0.15	-0.01	-0.04	0.14	-0.05	0.07	0.13	-0.04	0.05	0.14	-0.03
1420	1	0.09	0.17	0.06	0.16	0.06	0.02	0.02	0.07	0.03	0.10	0.00	0.01	-0.02	0.04	-0.03	0.03	0.03	0.07	0.02	0.07	0.01	0.13	-0.01	0.05
	2	0.41	0.92	0.21	0.82	0.23	0.11	0.07	0.58	0.14	0.55	0.11	0.00	0.10	0.45	0.05	-0.14	0.06	-0.02	0.00	0.37	0.03	0.07	0.08	0.04
	3	0.05	0.11	0.03	0.09	0.03	0.01	0.01	0.07	0.06	0.04	-0.03	-0.02	0.01	0.04	0.01	-0.02	0.01	0.00	0.02	0.04	0.02	0.01	-0.03	0.01
	4	0.34	0.60	0.15	0.54	0.33	0.09	0.09	-0.04	-0.03	-0.05	0.10	0.14	0.03	0.05	-0.02	0.00	0.09	0.01	0.03	0.04	0.01	-0.03	0.08	0.03
	5	0.13	0.24	0.05	0.21	0.13	0.05	0.02	0.05	0.05	0.05	0.05	0.03	0.01	0.06	0.01	-0.02	-0.03	0.06	0.08	0.02	-0.07	-0.05	0.08	0.02
	6	0.25	0.45	0.10	0.41	0.19	0.05	0.05	0.10	0.02	0.10	0.07	0.05	0.09	0.10	0.00	-0.03	0.08	0.01	0.04	0.10	0.01	0.02	0.05	0.03

TABLE 6.2: A correlation matrix for 1) the whole dataset, 2) the 7:00AM to 11:00AM dataset, 3) the 11:00AM to 3:00PM dataset, and 4) the 3:00PM to 7:00PM dataset showing the correlations with site 1161.

fluctuations do not appear to be connected to the geographical distance from the reference site (1161).

The correlation matrix for temporal segment 7:00AM to 11:00AM, as shown in Table 6.2, is notably different from the global correlation matrix. Input feature two, cars or vans, is still strongly correlated (average of 65%), however, the spread is bigger (47% to 76%), showing that some of the locations have their distinct patterns during morning rush hour, not shown in the global correlation. Furthermore, geographical location now has some influence on correlation, with the neighbouring RTSSs, 1414 and 1415, having a correlation of 74% and 62%. More interestingly, variable four, rigid goods, has gone from a good positive correlation to a negative correlation in six of the 12 sites, with a range of -18% to 20%. This makes logical sense, as the rigid good drivers would try to avoid the rush hour traffic. This is a contradiction to the strong positive correlation shown in the global correlation matrix.

The correlation matrix for temporal segment 11:00AM to 3:00PM, 6.2, shows that input feature two, cars and vans, has diminished more, with an average correlation of 37% and a bigger spread of 11 to 50. The geographical location seems to have a strong impact on the correlation, with the neighbouring RTSSs, 1414 and 1415, having 48% and 50% respectively. Furthermore, for input feature four, rigid goods, none of the 12 sites have a positive correlation, in stark contrast to the good positive correlation of the global correlation matrix.

The correlation matrix for temporal segment 3:00PM to 7:00PM, as shown in Table 6.2, changes again when compared to the global correlation matrix and the previous time segment matrix. Variable four has now gone from all negative correlations to all bar one now having a positive correlation, as shown in Figure 6.6. This is an interesting observation which deserves more investigation out of the scope of this thesis, however, it gives more evidence of the evolving nature of the input features.

In conclusion, the global correlation matrix was insufficient to explain the correlation across time and space for the road traffic data. The global correlation exhibits a strong correlation for input feature two, however, this appears to have a diminishing effect as the day progresses. Furthermore, there is a geographical element to the input features which the global correlation does not identify. Lastly, there is a negative correlation for input feature four during certain times of the day, as shown in Figure 6.6, that the global correlation matrix also failed to identify. This illustrates that static

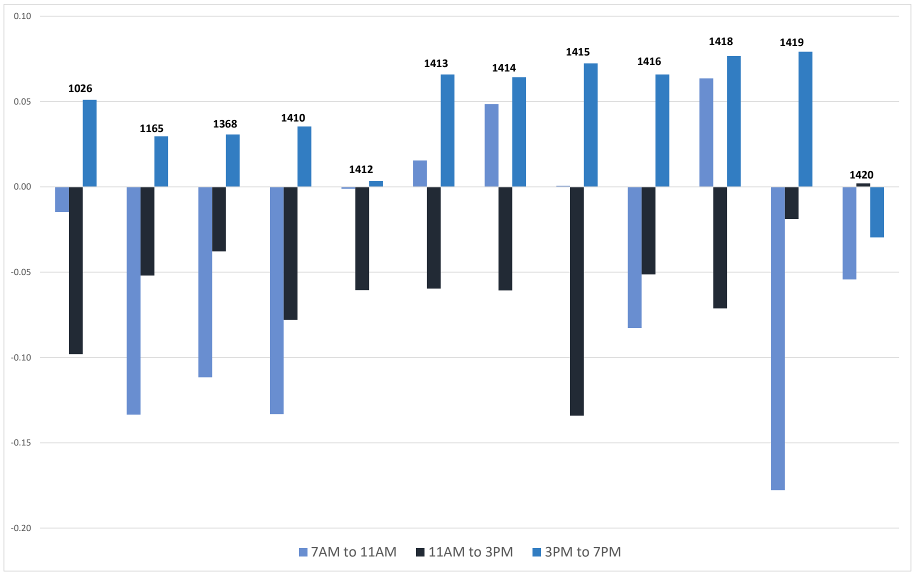

FIGURE 6.6: The dynamic nature of input feature four illustrated across the three time segments (7:00AM to 11:00PM, 11:00:PM to 3:00PM, and 3:00PM to 7:00PM) for each RTSS when correlated with the site of prediction (1161)

input features and statically trained models for the prediction of road traffic flow are inadequate to capture the complex and dynamic nature of the input features over space and time. A dynamic feature selection and prediction model is needed.

6.4.3 Research Question Two

To address research question two, can a novel dynamic exogenous feature filter method for the prediction of road traffic flow on a network improve prediction accuracy, the novel dynamic exogenous feature filter mechanism and three other models were trained and tested on the same dataset and their results were compared. The models were:

1. A state-of-the-art dynamic rolling window feature selection model.
2. The proposed local windows feature selection mechanism model.
3. A static feature selection model.
4. A model with no feature selection.

All four model architectures and parameters for experimentation were determined by using prior knowledge from the literature review and/or heuristics through grid search.

All weights and biases were randomly initialised and a dropout rate of 50% was used based on research by Zhao, Chen, Wu, *et al.* [33] and Srivastava, Hinton, Krizhevsky, *et al.* [92]. All other hyperparameters, such as the number of layers, nodes, and update window size, were determined through brute force by a random grid search to find the optimal design. The grid search searched through different architectural structures ranging from two to six layers (excluding any input and output layers) and nodes ranging from 12 to 20 with different hyperparameters to find the optimal design.

The different architectural structures and their variations were initially trained using the two months of training data, as described in Section 6.4.4. The models' weights and biases were then continually updated during the one-month test data through online learning (time series/rolling cross-validation). The Adamax optimiser [93] was used to optimise the models' learning rate based on the data's characteristics (a separate step size, known as the learning rate, for each parameter) during both the training and test phase. Each variation of the models' architectures was trained and tested 10 times and an average empirical error, based on the performance metric described in Section 6.4.4, was taken. The architecture that produced the lowest average empirical error was used for comparison.

6.4.4 Performance Metric

In order to compare and evaluate the accuracy of all four models the RMSE, as shown in Equation 2.3, and the STATS [122], as shown in Equation 4.15, will be used. There were no external constraints to take into consideration during experimentation. Therefore, when using the STATS performance metrics no upper or lower bounds were used and a priori equal weighting was given to the accuracy and training time ($wa = wb = 50$) to express lack of specific weighted preference.no upper or lower bounds were used during experimentation, and the accuracy and training time were given equal weighting ($w_a = w_b = 50$) .

6.4.5 Evaluation of the Proposed Novel Dynamic Exogenous Feature Filter for the Prediction of Road Traffic Flow

In table 6.3 the results for the no feature selection model show that it is the fastest, with a time score of 96.72%. This is due to no feature selection method being used

TABLE 6.3: Comparison of the feature filter methods

Feature Selection Model	**RMSE (%)**	**STATS**		
		Accuracy Score (%)	**Time Score (%)**	**Overall Score (%)**
None	12.82	87.18	96.72	91.95
Static	11.09	88.91	93.65	90.78
The Proposed Method	10.06	89.94	93.56	91.75
Dynamic Rolling Window	10.98	89.02	84.97	87.00

before or during the training of the model. However, it is also the worst performing model, with an RMSE of 12.82% (accuracy score of 87.18%). The duplication of similar input features may have caused the model to become too dependent on certain features leading to overfitting of the training data and therefore, struggling to generalise during the testing stage.

In the static feature selection model, the training dataset was used before the testing stage to determine which input features should be used. This model performed better than the model with no feature selection (as shown in Table 6.3) with an RMSE of 11.09% and an accuracy score of 88.91%. Removing duplicated input features has improved the prediction results, however, this has come at the cost to the computational load with a noticeable drop in the time score from the 'No Feature Selection' model from 96.72% to 93.65%.

The dynamic local windows feature filter mechanism, which uses three distinct periods: 1) morning rush hour peak, 2) midday trough, and 3) evening rush hour peak [124] to determine which input features should be used during these periods, improved the RMSE further to 10.06%, as shown in Table 6.3. Tailoring the input features to the different time windows has enabled the model to ensure that no duplicated traffic patterns are included in the training and test dataset and no relevant input features have been mistakenly disregarded because, as shown in Section 6.4.2, a global correlation can miss some correlations or assume correlations that are not present at certain periods of the day. Therefore, the 'local windows' mechanism was able to perform better. However, again this has come at a cost to the time, with a time score of 93.56%. Despite the extra time taken, it is still the overall best performing model with an overall score of 91.75%.

The dynamic rolling windows model, which uses an online rolling window to update the input feature's correlations and determine which input features should be used, performed worse than the dynamic local windows mechanism with an RMSE of 10.98% and an accuracy score of 89.02%. Furthermore, its time score, 84.97% was the lowest out of all the models by a large margin as the computational load of the rolling window and the regression model was quite high when compared to the other models.

In conclusion, the best performing model by a small margin was the model with no feature filter, with an overall score of 91.95%. However, its high score was solely due to its performance speed as it was also the least accurate model. The second best model, and the model with the highest accuracy score, was the dynamic windows model with an overall score of 91.75%. This model was the most accurate with an RMSE of 10.06% and it also scored well in the time score (93.56%). Therefore, overall the most successful model appears to be the dynamic local windows mechanism.

6.5 Summary

Accurate and timely road traffic flow prediction is vital for the management and planning of road traffic networks. One problem for accurate road traffic flow prediction is choosing which input features to use. To avoid overfitting many input features should be chosen, however, in the era of big data, there is an abundance of road traffic and other relevant data to select from. Choosing which input features to use is often down to the researchers' prior knowledge, or worse, best guess. This problem is compounded further by the very nature of road traffic flow in a network. Road traffic flow input features suffer from two main issues, firstly, the road traffic flow on a road network is highly correlated and therefore, adding more input features, which is usually used to avoid overfitting, can cause and exasperate overfitting and lead to a lack of generalisation during the testing phase. Secondly, road traffic flow suffers from concept drift, resulting in input features that were originally deemed vital becoming unnecessary, or worse, input features that were regarded as redundant and omitted may now be needed.

To address these issues we investigate the dynamic nature of the input features over time by examining their relationships using SRCC. Furthermore, a novel dynamic exogenous feature filter model was proposed and combined with a deep neural network for the prediction of road traffic flow. The proposed model was assessed on both accuracy and time taken and compared with three other models: 1) a model with no feature selection, 2) a model with static feature selection, and 3) a state-of-the-art dynamic rolling window feature selection.

The results from the input features analysis show that the global correlation matrix was insufficient to explain the correlation across time and space for the road traffic data. The local correlations showed that time and geographical local did affect correlation, and were able to more preciously show how the correlation of some input features diminish over time and how the geographical location was important at certain times of the day. Therefore, a dynamic feature selection and prediction model is needed. The results from the proposed model and the state-of-the-art model both performed better in terms of accuracy. The dynamic local windows feature selection mechanism performed the best with an RMSE of 10.06%, which was closely followed by the dynamic rolling window feature selection model, with an RMSE of 10.98%. The dynamic rolling window model did not do as well as expected; this may be due to the model struggling to retrain and adjust the weights when new input features were added during the online testing phase.

Both the proposed model and the dynamic rolling window model were, more computationally heavy when compared to a model with no feature selection filter and a model with a static feature selection filter and therefore, had higher time scores. The dynamic rolling window model in particular was very computationally heavy compared with the other three models, with a time score of 84.97%. Whereas, the dynamic local windows mechanism's time score was only marginally worse than the static feature selection model, with a time score of 93.56% and 93.65% respectively. Therefore, overall the proposed model, the dynamic local window mechanism, was the most accurate and efficient.

Chapter 7

Conclusion

Chapter Summary

The final chapter returns to the aims and objectives of this thesis, as shown in Section 1.2, and summaries the contributions of this research in each chapter. Furthermore, in Section **??** it discusses possible avenues for future development of the proposed framework.

7.1 Concluding Summary

Due to urbanisation, road traffic congestion has become a critical issue for most countries. In the UK alone, traffic congestion costs the economy £6.9 billion in 2019, and shows no sign of abating. Due to a lack of space in inner city areas, building new or widening existing roads is not possible. Therefore, better road traffic flow management is the only solution to avoid or mitigate road traffic congestion. As a result, research in the past two decades on Intelligent Transport Systems (ITS) has flourished. However, despite extensive research, an accurate and timely short-term road traffic flow prediction model for a road network is not yet available. Therefore, this thesis aimed to develop an accurate novel road traffic flow prediction model, with a particular focus on the short-term prediction of heterogeneous road traffic flow, for an urbanised road network.

In chapter two, we conducted a comprehensive literature review on road traffic flow prediction models (statistical, machine learning, and process-based) with applications to recurrent and non-recurrent road traffic flow, achieving objective one. It highlighted two main gaps in the literature; 1) there is no consensus on what methodology was most suitable for road traffic flow prediction, and 2) the datasets used for experimentation were inadequate. In general, the datasets were temporally and spatially small and only considered one input feature/data source.

Based on chapter two's findings, in chapter three, we performed a benchmark evaluation of existing machine learning models, partially achieving objective two. We examined their prediction accuracy and time-horizon sensitivity. We also examined different input feature settings (different classes of vehicles) to investigate how heterogeneous traffic flow can affect prediction accuracy, using a real dataset from Transport for Greater Manchester. The experimental results show that the ANN was most successful at predicting short-term road traffic flow on an urbanised road. Additionally, it was found that the inclusion of different classes of vehicles can improve prediction accuracy.

Therefore, in chapter four, we fully achieved objective two by examining three recurrent neural networks (a standard recurrent, a long short-term memory, and a gated recurrent unit) to determine how they perform on road traffic flow time-series data, based on a real dataset from Transport from Greater Manchester. We compared their accuracy, training time, and sensitivity to architectural change using a new performance metric we developed, Standardised Accuracy and Time Score (STATS), which standardises the accuracy and training time into a comparable score. The experimental results show that the gated recurrent unit performed the best and was most stable against architectural changes. Conversely, the long short-term memory was the least stable model.

In chapter five, we then moved on to investigating different magnitudes of temporal patterns in the road traffic flow dataset, such as short-term and long-term, to understand how contextual temporal data can improve prediction accuracy. This led to the development of a novel online dynamic temporal context neural network framework, achieving objective three. The framework dynamically determines how useful a temporal data segment (short and long-term temporal patterns) is for prediction, and weights it accordingly for use in the regression model, in real-time. Therefore, the framework can include short-term and relevant long-term patterns in the regression model, leading to improved prediction results. Using a real dataset, from Transport for Greater Manchester, containing daily, weekly, monthly and yearly data segments we performed a thorough evaluation. The experimental results show that short and long-term temporal patterns improved prediction accuracy. In addition, the proposed online dynamical framework improved prediction results by 10.8% when compared with a deep gated recurrent model.

In chapter six, we adapted the framework to include a road network to add spatial context. However, as input features are dynamic, in both time and space, we first examined the dynamic nature of the spatio-temporal input features' correlations. The results show that a global correlation was insufficient to describe the complex and dynamic relationships between the input features. The local correlations were able to identify additional geospatial and temporal relationships that the global correlation missed. Based on these findings, we developed a novel dynamic exogenous feature selection mechanism based on local windows and SRCC, achieving objective four. The proposed mechanism was compared to a state-of-the-art method, a dynamic rolling window feature filter method. We conducted a thorough experimental evaluation with a real dataset, from Transport for Greater Manchester, achieving objective five. The experimental results showed that the proposed filter feature mechanism was the most accurate, with an RMSE of 10.06%, closely followed by the dynamic rolling window feature filter model, with an RMSE of 10.98%. However, another advantage of the proposed feature the proposed feature filter mechanism was computationally much lighter than the rolling windows model.

In conclusion, we have achieved the main aim of the thesis, to develop an accurate novel road traffic flow prediction model, with a particular focus on the short-term prediction of heterogeneous road traffic flow on an urbanised road network, by creating a novel online dynamic temporal context neural network framework and a dynamic exogenous feature selection mechanism for the prediction of road traffic flow on an urbanised road network. Furthermore, the outcome of this thesis demonstrates the importance of input data. A prediction model can only be as good as its input data, therefore, the magnitude of the temporal, spatial, and input features' diversity will determine and restrict what temporal, spatial, and input feature cycles and patterns can be learnt. Providing relevant temporal, spatial, and diverse input features is vital for accurate road traffic flow prediction.

7.2 Limitations and Future Work

Due to the scope, restricted time and resources available during this research, there are limitations to this study and avenues that have not yet been investigated. These limitations and potential adaptations/extensions will now be explored in more detail. The limitations and adaptations/extensions, can be split into two categories; 1) data, and 2) prediction model.

7.2.1 Data

During this research real-life datasets, described in Section 3.2.4, provided by Transport for Greater Manchester were used for experimentation. The datasets contained three months of road traffic flow broken down into different vehicle classes, with a time horizon of five minutes. Although efforts were made to use temporally and spatially extensive datasets, one limitation of this thesis is their scope. Therefore, in this section, we will briefly detail the limitations of the datasets and how this research could be expanded upon if time permitted.

The main limitation of the datasets used is selection bias. The research only considers road traffic flow from Manchester, leading to limitations in the study's scope. Can we make generalisations that the proposed framework will work in other geographical locations? Driving style and behaviour varies from culture to culture [146]. Therefore, more representative datasets from other urbanised areas in the UK, or other countries, to test the proposed framework should be explored. Furthermore, their heterogeneous traffic flows should be analysed to quantify any cultural differences in road traffic flows.

Another limitation of the dataset used for experimental evaluation is the confounding parameters. The datasets contained only road traffic flow broken down into different vehicle classes, as shown in Table 3.1. Many studies have found that other input parameters, such as weather and social networks [147] can improve prediction accuracy. Furthermore, in this thesis we have shown how adding more contextual information (spatial, temporal, and input feature diversity) can improve prediction results. Therefore, adding more input parameter/context to the proposed framework, such as planned events, annual holidays, railway strikes, and even major events such as the COVID 'Stay at Home Order', should be investigated to improve accuracy.

7.2.2 Prediction Model

In this thesis, we have designed a deep learning framework specifically for predicting road traffic flow that yields good accuracy. However, there are limitations to the proposed framework and areas that can be improved.

The main limitation of the proposed framework is methodological limitations, which result from the selection bias detailed in Section 7.2.1. While we have thoroughly tested the novel framework, it has only been tested on road traffic flow from Manchester city centre. Therefore, can we generalise the results? Is Manchester city centre road network a good representation of the target population (any road network in an urbanised area)?

Continuing with methodological issues and datasets. The proposed framework, like most time-series regression models, relies on complete and accurate input data. In real-life situations, such data may not be available. Road traffic sensors are susceptible to malfunctions and breakdowns or may need recalibrating regularly. Therefore, the proposed framework should be extended further to deal with missing datasets and values if it is to have real-world applications.

Another limitation of this research is, due to continued research, ANNs (and regression models in general) is a rapidly expanding field, and new concepts and applications are continuously added. Therefore, in this section, we will briefly review alternative models or structural extensions/adaptations that could be explored to improve the proposed framework if time permitted.

Part of the novel online dynamic temporal context neural network framework was a CNN with a unique sliding window, as described in Section 5.3. The sliding window is pivotal to CNNs, as its size and shape dictate what temporal and/or spatial patterns can be learnt. One concept that could be explored is a dynamic window. The spatial dependencies in a road network change over time. During free flow, it would be advantageous to consider numerous non-neighbouring RTSSs from the road network to improve prediction accuracy. However, during times of severe congestion prediction results may be improved by only considering limited neighbouring RTSSs. Therefore, having a sliding window that can adapt in real-time to the state of the road would be an interesting concept to explore.

Another aspect of the framework that could be adapted is the ANN structure used to add a geospatial aspect to the data. The current proposed framework uses a CNN. CNNs operate on Euclidean data structures (1D or 2D matrices), however, road networks are not so uniform. Graph neural networks (GNNs) [148], unlike CNNs, are essentially a 'diffusion mechanism', meaning they are designed to spread information across a network. Therefore, exploring the expressive power of a graph to denote a road network (non-Euclidean data) should be investigated further to improve the proposed framework's prediction results.

Chapter 8

References

[1] Unknown, *INRIX Global Traffic Scorecard: Congestion cost UK economy 6.9 billion pounds in 2019*, Mar. 2020. [Online]. Available: https://www.automotiveworld.com/news-releases/inrix-global-traffic-scorecard-congestion-cost-uk-economy-6-9-billion-in-2019/.

[2] Department for Transport, *Statistical Release Minor Road Traffic Estimates Revisions*, Sep. 2020. [Online]. Available: https://www.gov.uk/government/.

[3] Y. Zhang, M. N. Smirnova, J. Ma, Z. Zhu, and N. N. Smirnov, "Freeway tunnel effect of travel time based-on a double lane traffic model," *International Journal of Transportation Science and Technology*, vol. 11, no. 2, pp. 360–380, Jun. 2022, ISSN: 20460449. DOI: 10.1016/J.IJTST.2021.05.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2046043021000368.

[4] F. Guo, R. Krishnan, and J. Polak, "Short-Term Traffic Prediction Under Normal and Abnormal Traffic Conditions on Urban Roads," in *Transportation Research Board 91st Annual Meeting, At Washington D.C.*, 2014. [Online]. Available: https://www.researchgate.net/publication/257942888_Short-Term_Traffic_Prediction_Under_Normal_and_Abnormal_Traffic_Conditions_on_Urban_Roads.

[5] J. Barros, M. Araujo, and R. J. F. Rossetti, "Short-term real-time traffic prediction methods: A survey," in *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, Jun. 2015, pp. 132–139, ISBN: 978-9-6331-3140-4. DOI: 10.1109/MTITS.2015.7223248. [Online]. Available: http://ieeexplore.ieee.org/document/7223248/.

[6] B. Sun, W. Cheng, P. Goswami, and G. Bai, "Flow-aware WPT k-nearest neighbours regression for short-term traffic prediction," in *2017 IEEE Symposium on Computers and Communications (ISCC)*, IEEE, Jul. 2017, pp. 48–53, ISBN: 978-1-5386-1629-1. DOI: 10.1109/ISCC.2017.8024503. [Online]. Available: http://ieeexplore.ieee.org/document/8024503/.

[7] F. Guo, "Short-term traffic prediction under normal and abnormal conditions," Ph.D. dissertation, Imperial College London, 2013. [Online]. Available: https://spiral.imperial.ac.uk/bitstream/10044/1/23661/1/Guo-F-2014-PhD-Thesis.pdf.

[8] G. E. P. Box and G. M. Jenkins, *Time series analysis: Forecasting and control*, Rev. ed. San Francisco : Holden-Day, 1976, p. 575, ISBN: 0816211043. [Online]. Available: https://searchworks.stanford.edu/view/875030.

[9] R. F. Engle, "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation," *Econometrica*, vol. 50, no. 4, p. 987, Jul. 1982, ISSN: 00129682. DOI: 10.2307/1912773. [Online]. Available: http://www.jstor.org/stable/1912773?origin=crossref.

[10] T. Bollerslev, "Generalized Autoregressive Conditional Heteroskedasticity," *Journal of Econometrics*, vol. 31, pp. 307–327, 1986. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.468.2892&rep=rep1&type=pdf.

[11] J. Peng, Y. Xu, and M. Wu, "Short-Term Traffic Flow Forecast Based on ARIMA-SVM Combined Model," in *International Conference on Green Intelligent Transportation System and Safety*, Springer, Singapore, 2023, pp. 287–300, ISBN: 9789811956140. DOI: 10.1007/978-981-19-5615-720/COVER. [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-19-5615-7_20.

[12] H. Yang, X. Li, W. Qiang, Y. Zhao, W. Zhang, and C. Tang, "A network traffic forecasting method based on SA optimized ARIMA–BP neural network," *Computer Networks*, vol. 193, p. 108 102, Jul. 2021, ISSN: 1389-1286. DOI: 10.1016/J.COMNET.2021.108102.

[13] M. Van Der Voort, M. Dougherty, and S. Watson, "Combining kohonen maps with arima time series models to forecast traffic flow," *Transportation Research Part C: Emerging Technologies*, vol. 4, no. 5, pp. 307–318, Oct. 1996, ISSN: 0968090X.

DOI: 10.1016/S0968-090X(97)82903-8. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0968090X97829038.

[14] D.-w. Xu, Y.-d. Wang, L.-m. Jia, Y. Qin, and H.-h. Dong, "Real-time road traffic state prediction based on ARIMA and Kalman filter," *Frontiers of Information Technology & Electronic Engineering*, vol. 18, no. 2, pp. 287–302, Feb. 2017, ISSN: 2095-9184. DOI: 10.1631/FITEE.1500381. [Online]. Available: http://link.springer.com/10.1631/FITEE.1500381.

[15] N. Deretić, D. Stanimirović, M. Al Awadh, N. Vujanović, and A. Djukić, "SARIMA Modelling Approach for Forecasting of Traffic Accidents," *Sustainability 2022, Vol. 14, Page 4403*, vol. 14, no. 8, p. 4403, Apr. 2022, ISSN: 2071-1050. DOI: 10.3390/SU14084403. [Online]. Available: https://www.mdpi.com/2071-1050/14/8/4403/htmhttps://www.mdpi.com/2071-1050/14/8/4403.

[16] B. M. Williams and L. A. Hoel, "Modeling and Forecasting Vehicular Traffic Flow as a Seasonal ARIMA Process: Theoretical Basis and Empirical Results," *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 664–672, Nov. 2003.

[17] S. V. Kumar and L. Vanajakshi, "Short-term traffic flow prediction using seasonal ARIMA model with limited input data," *European Transport Research Review*, vol. 7, no. 3, p. 21, Sep. 2015, ISSN: 1867-0717. DOI: 10.1007/s12544-015-0170-8. [Online]. Available: http://link.springer.com/10.1007/s12544-015-0170-8.

[18] B. Alsolami, R. Mehmood, and A. Albeshri, "Hybrid statistical and machine learning methods for road traffic prediction: A review and tutorial," *EAI/Springer Innovations in Communication and Computing*, pp. 115–133, 2020, ISSN: 25228609. DOI: 10.1007/978-3-030-13705-2{_}5/COVER/. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-13705-2_5.

[19] H. Dong, L. Jia, X. Sun, C. Li, and Y. Qin, "Road Traffic Flow Prediction with a Time-Oriented ARIMA Model," in *2009 Fifth International Joint Conference on INC, IMS and IDC*, IEEE, 2009, pp. 1649–1652, ISBN: 978-1-4244-5209-5. DOI: 10.1109/NCM.2009.224. [Online]. Available: http://ieeexplore.ieee.org/document/5331589/.

[20] G. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, Jan. 2003, ISSN: 09252312.

DOI: 10.1016/S0925-2312(01)00702-0. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0925231201007020.

[21] V. B. Gavirangaswamy, G. Gupta, A. Gupta, R. Agrawal, and N Carolina, "Assessment of ARIMA-based Prediction Techniques for Road-Traffic Volume," 2013. [Online]. Available: http://delivery.acm.org/10.1145/2540000/2536176/p246-gavirangaswamy.pdf?ip=149.170.92.57&id=2536176&acc=ACTIVE%20SERVICE&key=BF07A2EE685417C5%2E8670C7C518068584%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=1000216684&CFTOKEN=20725034&__acm__=1509363598_b965.

[22] B. W. Silverman and M. C. Jones, "E. Fix and J.L. Hodges (1951): An Important Contribution to Nonparametric Discriminant Analysis and Density Estimation: Commentary on Fix and Hodges (1951)," *International Statistical Review / Revue Internationale de Statistique*, vol. 57, no. 3, p. 233, Dec. 1989, ISSN: 03067734. DOI: 10.2307/1403796.

[23] B. Scholkopf, P. Simard, A. Smola, and V. Vapnikt, "Prior Knowledge in Support Vector Kernels," 1998. [Online]. Available: http://papers.nips.cc/paper/1457-prior-knowledge-in-support-vector-kernels.pdf.

[24] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, Dec. 1943. DOI: 10.1007/BF02478259. [Online]. Available: http://link.springer.com/10.1007/BF02478259.

[25] P. Cai, Y. Wang, G. Lu, P. Chen, C. Ding, and J. Sun, "A spatiotemporal correlative k-nearest neighbor model for short-term traffic multistep forecasting," *Transportation Research Part C: Emerging Technologies*, vol. 62, pp. 21–34, Jan. 2016. DOI: 10.1016/J.TRC.2015.11.002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X15003812.

[26] K.-R. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf, "An Introduction to Kernel-Based Learning Algorithms," *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 181–201, Mar. 2001. DOI: 10.1109/72.914517. [Online]. Available: http://ieeexplore.ieee.org/document/914517/.

[27] G. Lin, A. Lin, and D. Gu, "Using support vector regression and K-nearest neighbors for short-term traffic flow prediction based on maximal information coefficient," *Information Sciences*, vol. 608, pp. 517–531, Aug. 2022, ISSN: 0020-0255. DOI: 10.1016/J.INS.2022.06.090.

[28] W. Hu, L. Yan, K. Liu, and H. Wang, "A Short-term Traffic Flow Forecasting Method Based on the Hybrid PSO-SVR," *Neural Processing Letters*, vol. 43, no. 1, pp. 155–172, Feb. 2016. DOI: 10.1007/s11063-015-9409-6. [Online]. Available: http://link.springer.com/10.1007/s11063-015-9409-6.

[29] X. Ling, X. Feng, Z. Chen, Y. Xu, and H. Zheng, "Short-term traffic flow prediction with optimized Multi-kernel Support Vector Machine," in *2017 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, Jun. 2017, pp. 294–300, ISBN: 978-1-5090-4601-0. DOI: 10.1109/CEC.2017.7969326. [Online]. Available: http://ieeexplore.ieee.org/document/7969326/.

[30] P. Cao, F. Dai, G. Liu, J. Yang, and B. Huang, "A Survey of Traffic Prediction Based on Deep Neural Network: Data, Methods and Challenges," *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICS*, vol. 430, pp. 17–29, 2022, ISSN: 1867822X. DOI: 10.1007/978-3-030-99191-3{_}2/COVER/. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-99191-3_2.

[31] J. Jin, D. Rong, T. Zhang, *et al.*, "A GAN-Based Short-Term Link Traffic Prediction Approach for Urban Road Networks Under a Parallel Learning Framework," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 16 185–16 196, Feb. 2022, ISSN: 15580016. DOI: 10.1109/TITS.2022.3148358. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9713756.

[32] R. More, A. Mugal, S. Rajgure, R. B. Adhao, and V. K. Pachghare, "Road traffic prediction and congestion control using Artificial Neural Networks," in *2016 International Conference on Computing, Analytics and Security Trends (CAST)*, IEEE, Dec. 2016, pp. 52–57, ISBN: 978-1-5090-1338-8.

[33] Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen, and J. Liu, "LSTM network: a deep learning approach for short-term traffic forecast," *IET Intelligent Transport Systems*, vol. 11, no. 2, pp. 68–75, Mar. 2017.

[34] K. Cho, B. van Merrienboer, C. Gulcehre, *et al.*, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," Jun. 2014.

[35] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[36] J. Wang, L.-C. Yu, K. R. Lai, and X. Zhang, "Dimensional Sentiment Analysis Using a Regional CNN-LSTM Model," in *54th Annual Meeting of the Association for Computational Linguistics*, Berlin, Germany: Association for Computational Linguistics, 2016, pp. 225–230.

[37] Y. Wu, H. Tan, L. Qin, B. Ran, and Z. Jiang, "A hybrid deep learning based traffic flow prediction method and its understanding," *Transportation Research Part C: Emerging Technologies*, vol. 90, pp. 166–180, May 2018.

[38] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, IEEE, Nov. 2016, pp. 324–328.

[39] L. Zheng, C. Zhu, N. Zhu, T. He, N. Dong, and H. Huang, "Feature selection-based approach for urban short-term travel speed prediction," *IET Intelligent Transport Systems*, vol. 12, no. 6, pp. 474–484, Aug. 2018, ISSN: 1751956X. DOI: 10.1049/IET-ITS.2017.0059.

[40] M. Banko and E. Brill, "Mitigating the paucity-of-data problem," in *Proceedings of the first international conference on Human language technology research - HLT '01*, Morristown, NJ, USA: Association for Computational Linguistics, 2001, pp. 1–5.

[41] Z. H. Mir and F. Filali, "An adaptive Kalman filter based traffic prediction algorithm for urban road network," in *2016 12th International Conference on Innovations in Information Technology (IIT)*, IEEE, Nov. 2016, pp. 1–6, ISBN: 978-1-5090-5341-4. DOI: 10.1109/INNOVATIONS.2016.7880022. [Online]. Available: http://ieeexplore.ieee.org/document/7880022/.

[42] E. Kayacana, B. Ulutas, and O. Kaynaka, "Grey system theory-based models in time series prediction," *Expert Systems with Applications*, vol. 37, no. 2, pp. 1784–1789, Mar. 2010. DOI: 10.1016/J.ESWA.2009.07.064. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417409007258.

[43] K. Ramamohanarao, H. Xie, L. Kulik, *et al.*, "SMARTS: Scalable Microscopic Adaptive Road Traffic Simulator," *ACM Trans. Intell. Syst. Technol. Article*, vol. 8, no. 26, 2016. DOI: `10.1145/2898363`. [Online]. Available: `http://dx.doi.org/10.1145/2898363`.

[44] S. Suhas, V. V. Kalyan, M. Katti, B. V. A. Prakash, and C. Naveena, "A Comprehensive Review on Traffic Prediction for Intelligent Transport System," in *2017 International Conference on Recent Advances in Electronics and Communication Technology*, IEEE, Mar. 2017, pp. 138–143, ISBN: 978-1-5090-6701-5.

[45] Y. Wang, R. Jia, F. Dai, and Y. Ye, "Traffic Flow Prediction Method Based on Seasonal Characteristics and SARIMA-NAR Model," *Applied Sciences 2022, Vol. 12, Page 2190*, vol. 12, no. 4, p. 2190, Feb. 2022, ISSN: 2076-3417. DOI: `10.3390/APP12042190`. [Online]. Available: `https://www.mdpi.com/2076-3417/12/4/2190/htmhttps://www.mdpi.com/2076-3417/12/4/2190`.

[46] Z. Bartlett, L. Han, T. Nguyen, and P. Johnson, "A novel online dynamic temporal context neural network framework for the prediction of road traffic flow," *IEEE Access*, vol. 7, 2019, ISSN: 21693536. DOI: `10.1109/ACCESS.2019.2943028`.

[47] W. Huang, W. Jia, J. Guo, *et al.*, "Real-time prediction of seasonal heteroscedasticity in vehicular traffic flow series," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 10, pp. 3170–3180, Oct. 2018. DOI: `10.1109/TITS.2017.2774289`.

[48] X. Xiao, J. Yang, S. Mao, and J. Wen, "An improved seasonal rolling grey forecasting model using a cycle truncation accumulated generating operation for traffic flow," *Applied Mathematical Modelling*, vol. 51, pp. 386–404, Nov. 2017, ISSN: 0307-904X. DOI: `10.1016/J.APM.2017.07.010`.

[49] Y. Zhang, A. Haghani, and X. Zeng, "Component GARCH models to account for seasonal patterns and uncertainties in travel-time prediction," *IEEE Transactions on Intelligent Transportation Systems*, 2015, ISSN: 15249050. DOI: `10.1109/TITS.2014.2339097`.

[50] L. Zhang, Q. Liu, W. Yang, N. Wei, and D. Dong, "An Improved K-nearest Neighbor Model for Short-term Traffic Flow Prediction," *Procedia - Social and Behavioral Sciences*, vol. 96, pp. 653–662, Nov. 2013. DOI: `10.1016/J.SBSPRO.`

2013.08.076. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877042813022027.

[51] C. P. I. Van Hinsbergen, J. Lint, and F. M. Sanders, "Short Term Traffic Prediction Models," in *14th World Congress on Intelligent Transport Systems, ITS*, 2007. [Online]. Available: https://www.researchgate.net/publication/228882378_Short_Term_Traffic_Prediction_Models.

[52] W. Cui and D. Guo, "Vehicle delay series forecast based on trajectories of GPS tracked cabs," in *2015 23rd International Conference on Geoinformatics*, IEEE, Jun. 2015, pp. 1–7, ISBN: 978-1-4673-7663-1. DOI: 10.1109/GEOINFORMATICS.2015.7378575. [Online]. Available: http://ieeexplore.ieee.org/document/7378575/.

[53] A. Zhang, S. Song, J. Wang, and P. S. Yu, "Time Series Data Cleaning: From Anomaly Detection to Anomaly Repairing," Tsinghua University, Tech. Rep., 2017. [Online]. Available: http://delivery.acm.org/10.1145/3120000/3115410/p1046-zhang.pdf?ip=149.170.92.57&id=3115410&acc=ACTIVE%20SERVICE&key=BF07A2EE685417C5%2E8670C7C518068584%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=981410384&CFTOKEN=38026495&__acm__=1504695851_8e611740688ed.

[54] H. N. Akouemo and R. J. Povinelli, "Data Improving in Time Series Using ARX and ANN Models," *IEEE Transactions on Power Systems*, vol. 32, no. 5, pp. 3352–3359, Sep. 2017, ISSN: 0885-8950. DOI: 10.1109/TPWRS.2017.2656939. [Online]. Available: http://ieeexplore.ieee.org/document/7831434/.

[55] H. Yan, L. Fu, Y. Qi, D. J. Yu, and Q. Ye, "Robust ensemble method for short-term traffic flow prediction," *Future Generation Computer Systems*, vol. 133, pp. 395–410, Aug. 2022, ISSN: 0167-739X. DOI: 10.1016/J.FUTURE.2022.03.034. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0167739X22001157.

[56] E. Doğan, "Robust-LSTM: a novel approach to short-traffic flow prediction based on signal decomposition," *Soft Computing*, vol. 26, no. 11, pp. 5227–5239, Jun. 2022, ISSN: 14337479. DOI: 10.1007/S00500-022-07023-W/METRICS. [Online]. Available: https://link.springer.com/article/10.1007/s00500-022-07023-w.

[57] C. Goves, R. North, R. Johnston, and G. Fletcher, "Short Term Traffic Prediction on the UK Motorway Network Using Neural Networks," *Transportation Research Procedia*, vol. 13, pp. 184–195, Jan. 2016. DOI: 10.1016/J.TRPRO.2016.05.019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2352146516300199#!.

[58] A. Izabel, J Tostes, F. De, *et al.*, "From Data to Knowledge: City-wide Traffic Flows Analysis and Prediction Using Bing Maps," in *ACM SIGKDD International Workshop on Urban Computing Article No. 12*, 2013. [Online]. Available: https://www.cs.uic.edu/~urbcomp2013/papers/Paper%2015_Tostes.pdf.

[59] X. Liu, X. Kong, and Y. Li, "Collective Traffic Prediction with Partially Observed Traffic History using Location-Based Social Media," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management - CIKM '16*, New York, New York, USA: ACM Press, 2016, pp. 2179–2184, ISBN: 9781450340731. DOI: 10.1145/2983323.2983662. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2983323.2983662.

[60] Z. Zhu, B. Peng, C. Xiong, and L. Zhang, "Short-term traffic flow prediction with linear conditional Gaussian Bayesian network," *Journal of Advanced Transportation*, vol. 50, no. 6, pp. 1111–1123, Oct. 2016. DOI: 10.1002/atr.1392. [Online]. Available: http://doi.wiley.com/10.1002/atr.1392.

[61] R. Tahmasbi and S. M. Hashemi, "Modeling and Forecasting the Urban Volume Using Stochastic Differential Equations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 1, pp. 250–259, Feb. 2014. DOI: 10.1109/TITS.2013.2278614. [Online]. Available: http://ieeexplore.ieee.org/document/6588933/.

[62] G. A. Davis and N. L. Nihan, "Nonparametric Regression and Short-Term Freeway Traffic Forecasting," *Journal of Transportation Engineering*, vol. 117, no. 2, pp. 178–188, Mar. 1991. DOI: 10.1061/(ASCE)0733-947X(1991)117:2(178). [Online]. Available: http://ascelibrary.org/doi/10.1061/%28ASCE%290733-947X%281991%29117%3A2%28178%29.

[63] J.-L. Deng, *The Journal of grey system.* Sci-Tech Information Services, 1989, vol. 1, pp. 1–24. [Online]. Available: https://dl.acm.org/citation.cfm?id=90758.

[64] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960. [Online]. Available: https://www.cs.unc.edu/~welch/kalman/media/pdf/Kalman1960.pdf.

[65] M. S. Ahmed and A. R. Cook, "Analysis of Freeway Traffic Time-Series Data by Using Box-Jenkins Techniques," *Transportation Research Record*, vol. 722, pp. 1–9, 1997. [Online]. Available: http://onlinepubs.trb.org/Onlinepubs/trr/1979/722/722-001.pdf.

[66] M. Levin and Y.-D. Tsao, "On Forecasting FREEWAY Occupancies and Volumes (Abridgment)," *Transportation Research Record*, no. 773, 1980, ISSN: 0361-1981. [Online]. Available: https://trid.trb.org/view.aspx?id=167550.

[67] M. M. Hamed, H. R. Al-Masaeid, and Z. M. B. Said, "Short-Term Prediction of Traffic Volume in Urban Arterials," *Journal of Transportation Engineering*, vol. 121, no. 3, pp. 249–254, May 1995, ISSN: 0733-947X. DOI: 10.1061/(ASCE)0733-947X(1995)121:3(249). [Online]. Available: http://ascelibrary.org/doi/10.1061/%28ASCE%290733-947X%281995%29121%3A3%28249%29.

[68] A. Koesdwiady, R. Soua, and F. Karray, "Improving Traffic Flow Prediction With Weather Information in Connected Cars: A Deep Learning Approach," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 12, pp. 9508–9517, Dec. 2016.

[69] L. Lin, M. Ni, Q. He, J. Gao, and A. W. Sadek, "Modeling the Impacts of Inclement Weather on Freeway Traffic Speed," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2482, pp. 82–89, Sep. 2015, ISSN: 0361-1981. DOI: 10.3141/2482-11. [Online]. Available: http://trrjournalonline.trb.org/doi/10.3141/2482-11.

[70] N. Groschwitz and G. Polyzos, "A time series model of long-term NSFNET backbone traffic," in *International Conference on Communications - ICC/SUPERCOMM'94*, Orleans: IEEE, May 2002, pp. 1400–1404, ISBN: 0-7803-1825-0. DOI: 10.1109/ICC.1994.368876. [Online]. Available: http://ieeexplore.ieee.org/document/368876/.

[71] M. Lippi, M. Bertini, and P. Frasconi, "Short-Term Traffic Flow Forecasting: An Experimental Comparison of Time-Series Analysis and Supervised Learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 871–

882, Jun. 2013, ISSN: 1524-9050. DOI: 10.1109/TITS.2013.2247040. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6482260.

[72] T. Mai, B. Ghosh, and S. Wilson, "Multivariate Short-Term Traffic Flow Forecasting Using Bayesian Vector Autoregressive Moving Average Model," 2012. [Online]. Available: https://trid.trb.org/view.aspx?id=1130457.

[73] H. R. Kirby, S. M. Watson, and M. S. Dougherty, "Should we use neural networks or statistical models for short-term motorway traffic forecasting?" *International Journal of Forecasting*, vol. 13, pp. 43–50, 1997. [Online]. Available: http://ac.els-cdn.com/S0169207096006991/1-s2.0-S0169207096006991-main.pdf?_tid=b7eb957e-7d05-11e7-be3f-00000aacb360&acdnat=1502285097_3cdcabe90f30458291349f2c63d0210b.

[74] L. Zhao, X. Wen, Y. Wang, and Y. Shao, "A novel hybrid model of ARIMA-MCC and CKDE-GARCH for urban short-term traffic flow prediction," *IET Intelligent Transport Systems*, vol. 16, no. 2, pp. 206–217, Feb. 2022, ISSN: 1751-9578. DOI: 10.1049/ITR2.12138. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1049/itr2.12138https://onlinelibrary.wiley.com/doi/abs/10.1049/itr2.12138https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/itr2.12138.

[75] J.-L. Deng, "Control problems of grey systems," *Systems & Control Letters*, vol. 1, no. 5, pp. 288–294, Mar. 1982. DOI: 10.1016/S0167-6911(82)80025-X. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016769118280025X.

[76] H. Duan and G. Wang, "Partial differential grey model based on control matrix and its application in short-term traffic flow prediction," *Applied Mathematical Modelling*, vol. 116, pp. 763–785, Apr. 2023, ISSN: 0307-904X. DOI: 10.1016/J.APM.2022.12.012.

[77] B. Smith and M. Demetsky, "Short-term traffic flow prediction models-a comparison of neural network and nonparametric regression approaches," in *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, vol. 2, IEEE, 1994, pp. 1706–1709, ISBN: 0-7803-2129-4. DOI: 10.1109/ICSMC.1994.400094. [Online]. Available: http://ieeexplore.ieee.org/document/400094/.

[78] D.-w. Xu, Y.-d. Wang, L.-m. Jia, G.-j. Zhang, and H.-f. Guo, "Real-time road traffic states estimation based on kernel-KNN matching of road traffic spatial characteristics," *Journal of Central South University*, vol. 23, no. 9, pp. 2453–2464, Sep. 2016, ISSN: 2095-2899. DOI: 10.1007/s11771-016-3304-9. [Online]. Available: http://link.springer.com/10.1007/s11771-016-3304-9.

[79] A. Mallek, D. Klosa, and C. Buskens, "Enhanced K-Nearest Neighbor Model For Multi-steps Traffic Flow Forecast in Urban Roads," *IEEE International Smart Cities Conference (ISC2)*, pp. 1–5, Nov. 2022. DOI: 10.1109/ISC255366.2022.9921897. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9921897.

[80] D. C. Gazis and C. H. Knapp, "On-Line Estimation of Traffic Densities from Time-Series of Flow and Speed Data," *Transportation Science*, vol. 5, no. 3, pp. 283–301, Aug. 1971. DOI: 10.1287/trsc.5.3.283. [Online]. Available: http://pubsonline.informs.org/doi/abs/10.1287/trsc.5.3.283.

[81] I. Okutani and Y. J. Stephanedes, "Dynamic prediction of traffic volume through Kalman filtering theory," *Transportation Research Part B: Methodological*, vol. 18, no. 1, pp. 1–11, Feb. 1984. DOI: 10.1016/0191-2615(84)90002-X. [Online]. Available: http://www.sciencedirect.com/science/article/pii/019126158490002X.

[82] T. Zhou, D. Jiang, Z. Lin, G. Han, X. Xu, and J. Qin, "Hybrid dual Kalman filtering model for short-term traffic flow forecasting," *IET Intelligent Transport Systems*, vol. 13, no. 6, pp. 1023–1032, Jun. 2019, ISSN: 1751-9578. DOI: 10.1049/IET-ITS.2018.5385. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1049/iet-its.2018.5385https://onlinelibrary.wiley.com/doi/abs/10.1049/iet-its.2018.5385https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/iet-its.2018.5385.

[83] K. R. Müller, A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik, "Predicting time series with support vector machines," in Springer, Berlin, Heidelberg, 1997, pp. 999–1004. DOI: 10.1007/BFb0020283. [Online]. Available: http://link.springer.com/10.1007/BFb0020283.

[84] M. T. Asif, J. Dauwels, Chong Yang Goh, *et al.*, "Spatiotemporal Patterns in Large-Scale Traffic Speed Prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 2, pp. 794–804, Apr. 2014. DOI: 10.1109/TITS.

2013.2290285. [Online]. Available: http://ieeexplore.ieee.org/document/6678316/.

[85] W. Jiang and J. Luo, "Graph neural network for traffic forecasting: A survey," *Expert Systems with Applications*, vol. 207, p. 117 921, Nov. 2022, ISSN: 0957-4174. DOI: 10.1016/J.ESWA.2022.117921.

[86] A. S. Miller, B. H. Blott, and T. K. hames, "Review of neural network applications in medical imaging and signal processing," *Medical & Biological Engineering & Computing*, vol. 30, no. 5, pp. 449–464, Sep. 1992. DOI: 10.1007/BF02457822. [Online]. Available: http://link.springer.com/10.1007/BF02457822.

[87] C. Wang, S. Xu, J. Liu, J. Yang, and C. Liu, "Building an improved artificial neural network model based on deeply optimizing the input variables to enhance rutting prediction," *Construction and Building Materials*, vol. 348, p. 128 658, Sep. 2022, ISSN: 0950-0618. DOI: 10.1016/J.CONBUILDMAT.2022.128658.

[88] J. Barcelo, D. Garcia, and J. Perarnau, *Methodological Notes on Combining Macro, Meso and Micro Models for Transportation Analysis*, 2005. [Online]. Available: https://www.researchgate.net/publication/242154775_METHODOLOGICAL_NOTES_ON_COMBINING_MACRO_MESO_AND_MICRO_MODELS_FOR_TRANSPORTATION_ANALYSIS.

[89] K. Irawan, R. Yusuf, and A. S. Prihatmanto, "A Survey on Traffic Flow Prediction Methods," in *6th International Conference on Interactive Digital Media, ICIDM 2020*, Bandung, Indonesia: Institute of Electrical and Electronics Engineers Inc., Dec. 2020, ISBN: 9781728149288. DOI: 10.1109/ICIDM51048.2020.9339675.

[90] T. D. Toan and V. H. Truong, "Support vector machine for short-term traffic flow prediction and improvement of its model training using nearest neighbor approach," *Transportation Research Record*, vol. 2675, no. 4, pp. 362–373, Dec. 2021, ISSN: 21694052. DOI: 10.1177/0361198120980432/ASSET/IMAGES/LARGE/10.1177{_}0361198120980432-FIG2.JPEG. [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/0361198120980432.

[91] V. Cherkassky and Y. Ma, "Practical selection of SVM parameters and noise estimation for SVM regression.," *Neural networks : the official journal of the International Neural Network Society*, vol. 17, no. 1, pp. 113–26, Jan. 2004. DOI: 10.1016/S0893-6080(03)00169-2. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/14690712.

[92] N. Srivastava, G. Hinton, A. Krizhevsky, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[93] D. P. Kingma and J. L. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference on Learning Representations*, San Diego: International Conference on Learning Representations, ICLR, Dec. 2014. [Online]. Available: https://arxiv.org/abs/1412.6980v9.

[94] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006. DOI: 10.1162/NECO.2006.18.7.1527.

[95] R. Dechter, "Learning While Searching in Constraint Satisfaction Problems," in *Proceedings of the 5th National Conference on Artificial Intelligence*, Philadelphia, PA: ResearchGate, 1986, Volume 1: Science.

[96] C. Cortes, "Support-Vector Networks," Tech. Rep., 1995, pp. 273–297.

[97] W. Huang, G. Song, H. Hong, and K. Xie, "Deep Architecture for Traffic Flow Prediction: Deep Belief Networks With Multitask Learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2191–2201, Oct. 2014.

[98] J. Schmidhuber, "Deep Learning in Neural Networks: An Overview," *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015.

[99] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic Flow Prediction With Big Data: A Deep Learning Approach," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–9, 2014.

[100] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Short-term traffic forecasting: Where we are and where we are going," *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 3–19, Jun. 2014.

[101] M. I. Jordan, "Serial Order: A Parallel Distributed Processing Approach," *Advances in Psychology*, vol. 121, pp. 471–495, Jan. 1997.

[102] Z. Xu, Y. Kang, Y. Cao, and L. Yue, "Residual Autoencoder-LSTM for City Region Vehicle Emission Pollution Prediction," in *2018 IEEE 14th International Conference on Control and Automation*, IEEE, Jun. 2018, pp. 811–816, ISBN: 978-1-5386-6089-8.

[103] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to Forget: Continual Prediction with LSTM," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, Oct. 2000, ISSN: 0899-7667. DOI: `10.1162/089976600300015015`. [Online]. Available: `http://direct.mit.edu/neco/article-pdf/12/10/2451/814643/089976600300015015.pdf`.

[104] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A Novel Connectionist System for Unconstrained Handwriting Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, May 2009.

[105] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM networks," in *IEEE International Joint Conference on Neural Networks.2005*, vol. 4, IEEE, 2005, pp. 2047–2052, ISBN: 0-7803-9048-2.

[106] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. DOI: `10.1162/NECO.1997.9.8.1735`.

[107] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A Search Space Odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222 –2232, Mar. 2017.

[108] H. Shi, M. Xu, and R. Li, "Deep Learning for Household Load Forecasting A Novel Pooling Deep RNN," *IEEE Transactions on Smart Grid*, vol. 9, no. 5, pp. 5271–5280, Sep. 2018.

[109] D. Zhang and M. R. Kabuka, "Combining weather condition data to predict traffic flow: a GRU-based deep learning approach," *IET Intelligent Transport Systems*, vol. 12, no. 7, pp. 578–585, Sep. 2018.

[110] Z. Zhang, Y. Zhao, X. Liao, *et al.*, "Deep learning in omics: a survey and guideline," *Briefings in Functional Genomics*, Sep. 2018.

[111] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Deep Bidirectional and Unidirectional LSTM Recurrent Neural Network for Network-wide Traffic Speed Prediction," in *International Workshop on Urban Computing (UrbComp)*, Nova Scotia, Aug. 2017. [Online]. Available: `https://arxiv.org/abs/1801.02143v2`.

[112] H.-F. Yang, T. S. Dillon, and Y.-P. P. Chen, "Optimized Structure of the Traffic Flow Forecasting Model With a Deep Learning Approach," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2371–2381, Oct. 2017.

[113] N. G. Polson and V. O. Sokolov, "Deep learning for short-term traffic flow prediction," *Transportation Research Part C: Emerging Technologies*, vol. 79, pp. 1–17, Jun. 2017.

[114] X. Yin, G. Wu, J. Wei, Y. Shen, H. Qi, and B. Yin, "Deep Learning on Traffic Prediction: Methods, Analysis and Future Directions," *IEEE Transactions on Intelligent Transportation Systems*, Jun. 2021, ISSN: 15580016. DOI: `10.1109/TITS.2021.3054840`.

[115] N. Ranjan, S. Bhandari, H. P. Zhao, H. Kim, and P. Khan, "City-wide traffic congestion prediction based on CNN, LSTM and transpose CNN," *IEEE Access*, vol. 8, pp. 81 606–81 620, 2020, ISSN: 21693536. DOI: `10.1109/ACCESS.2020.2991462`.

[116] Peris, M. Bolaños, P. Radeva, and F. Casacuberta, *Video Description Using Bidirectional Recurrent Neural Networks*. Springer, Cham, 2016, pp. 3–11.

[117] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network Traffic Classifier With Convolutional and Recurrent Neural Networks for Internet of Things," *IEEE Access*, vol. 5, pp. 18 042–18 050, 2017.

[118] S. Narmadha and V. Vijayakumar, "Spatio-Temporal vehicle traffic flow prediction using multivariate CNN and LSTM model," *Materials Today: Proceedings*, May 2021, ISSN: 2214-7853. DOI: `10.1016/J.MATPR.2021.04.249`.

[119] M. E. Shaik, M. M. Islam, and Q. S. Hossain, "A review on neural network techniques for the prediction of road traffic accident severity," *Asian Transport Studies*, vol. 7, p. 100 040, Jan. 2021, ISSN: 2185-5560. DOI: `10.1016/J.EASTSJ.2021.100040`.

[120] T. Zhou, G. Han, X. Xu, *et al.*, "δ-agree AdaBoost stacked autoencoder for short-term traffic flow forecasting," *Neurocomputing*, vol. 247, pp. 31–38, Jul. 2017.

[121] Z. Bartlett, L. Han, T. Nguyen, and P. Johnson, "A Machine Learning Based Approach for the Prediction of Road Traffic Flow on Urbanised Arterial Roads," in *16th International Conference on Smart City 2018*, 2019.

[122] Z. Bartlett, L. Han, T. T. Nguyen, and P. Johnson, "Prediction of Road Traffic Flow Based on Deep Recurrent Neural Networks," in *The 5th IEEE Smart World Congress*, Leicester, UK: IEEE, 2019.

[123] H. Peng, H. Wang, B. Du, *et al.*, "Spatial temporal incidence dynamic graph neural networks for traffic flow forecasting," *Information Sciences*, vol. 521, pp. 277–290, Jun. 2020, ISSN: 00200255. DOI: 10.1016/j.ins.2020.01.043. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0020025520300451.

[124] T. Cheng, J. Haworth, and J. Wang, "Spatio-temporal autocorrelation of road network data," *Journal of Geographical Systems : Geographical Information, Analysis, Theory, and Decision*, vol. 14, no. 4, pp. 389–413, 2012, ISSN: 1435-5930. DOI: 10.1007/s10109-011-0149-5.

[125] S. Wang, J. Zhao, C. Shao, C. D. Dong, and C. Yin, "Truck Traffic Flow Prediction Based on LSTM and GRU Methods with Sampled GPS Data," *IEEE Access*, vol. 8, pp. 208 158–208 169, 2020, ISSN: 21693536. DOI: 10.1109/ACCESS.2020.3038788.

[126] Y. Wang and C. Jing, "Spatiotemporal Graph Convolutional Network for Multi-Scale Traffic Forecasting," *ISPRS International Journal of Geo-Information*, vol. 11, no. 2, p. 102, Feb. 2022, ISSN: 2220-9964. DOI: 10.3390/IJGI11020102. [Online]. Available: https://www.mdpi.com/2220-9964/11/2/102/htmhttps://www.mdpi.com/2220-9964/11/2/102.

[127] A. Ermagun and D. Levinson, "Spatiotemporal Short-term Traffic Forecasting using the Network Weight Matrix and Systematic Detrending," in *f Transportation Research Board 97th annual meeting*, Washington, USA: Transportation Research Board, 2018, pp. 14–14. [Online]. Available: https://ideas.repec.org/p/nex/wpaper/shorttermtrafficforecasting.html.

[128] J. Ou, J. Xia, Y. J. Wu, and W. Rao, "Short-term traffic flow forecasting for urban roads using data-driven feature selection strategy and bias-corrected random forests," *Transportation Research Record*, vol. 2645, no. 1, pp. 157–167, 2017, ISSN: 21694052. DOI: 10.3141/2645-17.

[129] X. Chen, Z. Wei, X. Liu, Y. Cai, Z. Li, and F. Zhao, "Spatiotemporal variable and parameter selection using sparse hybrid genetic algorithm for traffic flow forecasting:" *International Journal of Distributed Sensor Networks*, vol. 13, no. 6, pp. 1–14, Jun. 2017, ISSN: 15501477. DOI: 10.1177/1550147717713376. [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/1550147717713376.

[130] R. Tibshirani, "Regression Shrinkage and Selection Via the Lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, Jan. 1996, ISSN: 2517-6161. DOI: 10.1111/J.2517-6161.1996.TB02080.X. [Online]. Available: https://rss.onlinelibrary.wiley.com/doi/10.1111/j.2517-6161.1996.tb02080.x.

[131] J. P. Li, Hao Liu, Lin Xiong, *et al.*, "Graph CNNs for Urban Traffic Passenger Flows Prediction," in *2018 IEEE SmartWorld, Ubiquitous Intelligence and Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People and Smart City Innovations, SmartWorld/UIC/ATC/ScalCom/CBDCom/IoP/SCI 2018*, Institute of Electrical and Electronics Engineers Inc., 2018, pp. 29–36. [Online]. Available: https://www.researchgate.net/publication/329473709_Graph_CNNs_for_Urban_Traffic_Passenger_Flows_Prediction.

[132] Z. Li, S. Jiang, L. Li, and Y. Li, "Building sparse models for traffic flow prediction: an empirical comparison between statistical heuristics and geometric heuristics for Bayesian network approaches," *Transportmetrica B: Transport Dynamics*, vol. 7, no. 1, pp. 107–123, Dec. 2017, ISSN: 21680582. DOI: 10.1080/21680566.2017.1354737. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/21680566.2017.1354737.

[133] Q. Cao, G. Ren, and D. Li, "Multiple Spatio-temporal Scales Traffic Forecasting Based on Deep Learning Approach," in *Compendium of Papers of the Transportation Research Board 97th Annual Meeting*, Washington, USA: Transportation Board, 2018, pp. 19–19. [Online]. Available: https://trid.trb.org/view/1495400.

[134] D. Pavlyuk, "Feature selection and extraction in spatiotemporal traffic forecasting: a systematic literature review," *European Transport Research Review*, vol. 11, no. 1, pp. 1–19, Dec. 2019, ISSN: 18668887. DOI: 10.1186/S12544-019-0345-9/FIGURES/7. [Online]. Available: https://link.springer.com/articles/10.1186/s12544-019-0345-9.

[135] K. Pearson, "Notes on Regression and Inheritance in the Case of Two Parents," *Proceedings of the Royal Society of London*, vol. 58, no. LVIII, pp. 240–242, Apr. 1895. [Online]. Available: https://books.google.co.uk/books?id=60aL0zlT-90C&pg=PA240&redir_esc=y#v=onepage&q&f=false.

[136] K. Pearson, "On Lines and Planes of Closest Fit to Systems of Points in Space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, Nov. 1901. DOI: 10.1080/14786440109462720. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/14786440109462720.

[137] C Spearman, "The Proof and Measurement of Association Between Two Things," *American Journal of Psychology*, vol. 15, no. 1, pp. 72–101, 1904.

[138] W. Xiao, "Novel Online Algorithms for Nonparametric Correlations with Application to Analyze Sensor Data," in *2019 IEEE International Conference on Big Data*, Los Angeles, CA, USA: Institute of Electrical and Electronics Engineers Inc., Dec. 2019, pp. 404–412, ISBN: 9781728108582. DOI: 10.1109/BIGDATA47090.2019.9006483.

[139] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, Jan. 2014, ISSN: 0045-7906. DOI: 10.1016.

[140] L. Xie, Z. Li, Y. Zhou, Y. He, and J. Zhu, "Computational diagnostic techniques for electrocardiogram signal analysis," *Sensors (Switzerland)*, vol. 20, no. 21, pp. 1–32, Nov. 2020, ISSN: 14248220. DOI: 10.3390/S20216318.

[141] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001, ISSN: 1573-0565. DOI: 10.1023/A:1010933404324. [Online]. Available: https://link.springer.com/article/10.1023/A:1010933404324.

[142] J. H. J. H. Holland, *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence*. MIT Press, 1992, p. 211, ISBN: 9780262082136.

[143] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene Selection for Cancer Classification using Support Vector Machines," *Machine Learning 2002 46:1*, vol. 46, no. 1, pp. 389–422, 2002, ISSN: 1573-0565. DOI: 10.1023/A:1012487302797. [Online]. Available: https://link.springer.com/article/10.1023/A:1012487302797.

[144] D. E. Goldberg and J. H. Holland, "Genetic Algorithms and Machine Learning," *Machine Learning*, vol. 3, no. 2, pp. 95–99, Oct. 1988, ISSN: 1573-0565. DOI: 10.1023/A:1022602019183. [Online]. Available: https://link.springer.com/article/10.1023/A:1022602019183.

[145] X. Zhou, H. Hong, X. Xing, K. Bian, K. Xie, and M. Xu, "Discovering spatio-temporal dependencies based on time-lag in intelligent transportation data," *Neurocomputing*, vol. 259, pp. 76–84, Oct. 2017, ISSN: 0925-2312. DOI: 10.1016/J.NEUCOM.2016.06.084.

[146] T. Özkan, T. Lajunen, J. E. Chliaoutakis, D. Parker, and H. Summala, "Cross-cultural differences in driving behaviours: A comparison of six countries," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 9, no. 3, pp. 227–242, May 2006, ISSN: 1369-8478. DOI: 10.1016/J.TRF.2006.01.002.

[147] A. Essien, I. Petrounias, P. Sampaio, and S. Sampaio, "A deep-learning model for urban traffic flow prediction with traffic events mined from twitter," *World Wide Web*, vol. 24, no. 4, pp. 1345–1368, Jul. 2021, ISSN: 15731413. DOI: 10.1007/S11280-020-00800-3/TABLES/4. [Online]. Available: https://link.springer.com/article/10.1007/s11280-020-00800-3.

[148] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 2, IEEE, 2005, pp. 729–734, ISBN: 0-7803-9048-2. DOI: 10.1109/IJCNN.2005.1555942. [Online]. Available: http://ieeexplore.ieee.org/document/1555942/.

Glossary

catastrophic forgetting The lose of long-term temporal patterns in the training data through the continual updating of the ANN's weights and biases. 72

concept drift When the probability distributions of the input features change over time. 90

density The average number of vehicles on a section of road at a specified point in time. Usually given in vehicles per mile (VPM). 9

endogenous To be derived internally. 96

exogenous To be derived externally. 95

headway The time between two vehicles as they pass a specified point in the road. Usually given in seconds. 9

heterogeneous Diverse in contents and nature. 100

heteroscedastic The variance around the regression line is not evenly distributed and therefore, not the same for all predictor values (x). 100

homoscedastic The variance around the regression line is evenly distributed and therefore, the same for all predictor values (x). 10

input features The individual independent variables that act as the input in the traffic flow prediction models. 8

iteration Used to described the number of times all data points within a batch have passed through the model. 10

non-parametric No assumptions of the form or parameters of the frequency distribution have been made. 15

occupancy The percentage of time a specified section of the road is occupied by a vehicle. 9

outlier A data point which is more than two standard deviations away from the mean. 10

performance metric Measurable data used to evaluate the performance of the road traffic flow prediction models. 11

periodicity The tendency to recur at regular temporal intervals.. 15

robust The performance of the machine learning model is stable after adding some noise to the dataset. 10

saturation point The maximum number of vehicles the junction/road/network is able to handle before the traffic flow is impeded. 1

seasonality A repeated daily/weekly/monthly/yearly pattern within the road traffic flow data. 9

state The classification of the road traffic flow into three different categories, 1) free-flow, 2) saturated, and 3) jammed. Other categories have been suggested. 9

time horizon The change in time from one point to the next (t to $t+1$) in the historical data or prediction. 8, 13

timestep The time interval between one data point to the next.. 8, 13

traffic flow The number of vehicles that pass a specified point in the road during a time-step. 9

volume The number of vehicles on a section of road during one time-step. 9

Index