Short term traffic speed prediction on a large road network

TITING CUI
Short term traffic speed prediction on a large road network

TITING CUI
Abstract

Traffic flow speed prediction has been an important element in the application of intelligent transportation system (ITS). The timely and accurate traffic flow speed prediction can be utilized to support the control, management, and improvement of traffic conditions. In this project, we investigate the short term traffic flow speed prediction on a large highway network. To eliminate the vagueness, we first give a formal mathematical definition of traffic flow speed prediction problem on a road network. In the last decades, traffic flow prediction research has been advancing from the theoretically well established parametric methods to nonparametric data-driven algorithms, like the deep neural networks. In this research, we give a detailed review of the state-of-art prediction models appeared in the literature.

However, we find that the road networks are rather small in most of the literature, usually hundreds of road segments. The highway network in our project is much larger, consists of more than eighty thousand road segments, which makes it almost impossible to use the models in the literature directly. Therefore, in this research, we employ the time series clustering method to divide the road network into different disjoint regions. After that, several prediction models include historical average (HA), univariate and vector Autoregressive Integrated Moving Average model (ARIMA), support vector regression (SVR), Gaussian process regression (GPR), Stacked Autoencoders (SAEs), long short-term memory neural networks (LSTM) are selected to do the prediction on each region. We give a performance analysis of selected models at the end of the thesis.

Keywords: Traffic flow speed prediction; time series clustering; ARIMA; Gaussian process regression; support vector regression; Stacked Autoencoders; long short-term memory neural network
Sammanfattning


Vi finner dock att vägnätet är ganska litet i de flesta av litteraturen, vanligtvis hundratals vägsegment. Motorvägsnätverket i vårt projekt är mycket större, består av mer än 80 tusen vägsegment, vilket gör det nästan omöjligt att direkt använda modellerna i litteraturen. Därför använder vi i tidserien klustermetoden för att dela upp vägnätet i olika åtskilja regioner. Därefter innehåller flera prediktionsmodeller historisk medelvärde (HA), univariata och vector Autoregressive Integrated Moving Average-modellen (ARIMA), stödvektorregression (SVR), Gaussian processregression (GPR), Staplade Autoenkodare (SAEs) neurala nätverk (LSTM) väljs för att göra förutsägelsen för varje region. Vi ger en prestationsanalys av utvalda modeller i slutet av avhandlingen.

Nyckelord: Prognos för trafikflödeshastighet; tidsserie clustering; ARIMA; Gaussisk processregression; stöd vektor regression; Staplade autokodrar; lång ko- rttidsminne neuralt nätverk
Acknowledgements

I would like to thank Professor Pierre Nyquist, my master thesis adviser at KTH. Next, I would like to thank my supervisor Professor Minghua Chen at the Chinese University of Hong Kong for his constructive conversations and criticism during this project. Prof. Chen always does his best efforts to help me improve my understanding of the subject with great patience. I would like to thank the Ph.D. student, Wenjie Xu, for his inspiring discussion and engineering work. At last, I would like to thank all the other people in the DREAMS Lab at the Chinese University of Hong Kong.
Contents

1 Introduction .................................................. 9
   1.1 Background ........................................... 9
   1.2 Motivation ........................................... 10
   1.3 Problem definition .................................... 10
   1.4 Objectives and Research Scope ....................... 11
   1.5 Thesis outline ....................................... 11

2 Literature review: traffic prediction approaches .......... 13
   2.1 Naive approaches ..................................... 13
   2.2 Parametric approaches ................................ 14
      2.2.1 Linear regression ................................ 14
      2.2.2 ARIMA .......................................... 15
      2.2.3 Kalman filter .................................... 17
   2.3 Nonparametric approaches .............................. 18
      2.3.1 K nearest neighbors .............................. 18
      2.3.2 Support vector machine ......................... 18
      2.3.3 Gaussian regression .............................. 20
      2.3.4 Bayesian network ............................... 20
      2.3.5 Neural Network .................................. 21
   2.4 Hybrid approaches .................................... 23

3 Methodology ................................................ 25
   3.1 Time series clustering ................................ 25
      3.1.1 Shape-based distances .......................... 25
      3.1.2 Feature-based distances ......................... 28
      3.1.3 Clustering methods ............................. 29
   3.2 Prediction models ..................................... 29
      3.2.1 Clustering ........................................ 29
      3.2.2 Stacked Autoencoders ......................... 31
      3.2.3 LSTM Recurrent Neural Network ............... 32
<table>
<thead>
<tr>
<th>4</th>
<th>Experiment</th>
<th>34</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Data set</td>
<td>34</td>
</tr>
<tr>
<td>4.2</td>
<td>Evaluation metrics</td>
<td>36</td>
</tr>
<tr>
<td>4.3</td>
<td>Result</td>
<td>36</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5</th>
<th>Conclusion</th>
<th>39</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Conclusion</td>
<td>39</td>
</tr>
<tr>
<td>5.2</td>
<td>Future work</td>
<td>40</td>
</tr>
</tbody>
</table>
List of Figures

2.1 SVM with nonlinear transformation .............................................. 19
2.2 System architecture for the DCRNN designed for spatiotemporal traffic forecasting. The historical time series are fed into an encoder whose final states are used to initialize the decoder. The decoder makes predictions based on either previous ground truth or the model output [1] ........................................ 22
2.3 The architecture of STANN with two components: the encoder for modeling spatio-temporal dependencies and the decoder for multi-step traffic prediction [2] ........................................ 23
3.1 Structure of an autoencoder ......................................................... 31
3.2 Deep architecture of SAEs model ................................................. 32
3.3 Unfold structure of RNN ............................................................. 33
3.4 Graphical representation of an LSTM unit with peephole connections .................................................... 33
4.2 Sample speed data for one road segment ........................................ 35
4.3 The procedure of traffic flow speed prediction .................................... 35
4.4 Sample ACF and PACF for one edge .............................................. 36
4.5 Sample prediction of LSTM .......................................................... 38
## List of Tables

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Most commonly used (and effective) distance measures</td>
<td></td>
</tr>
<tr>
<td>4.1</td>
<td>Prediction performance of selected models</td>
<td></td>
</tr>
<tr>
<td>4.2</td>
<td>Performance comparison for DCRNN and GCRNN on the METRA-LA dataset</td>
<td></td>
</tr>
</tbody>
</table>

26

37

37
Chapter 1

Introduction

1.1 Background

The traffic flow prediction has been long regarded as a fundamental problem in an intelligent transportation system (ITS). Accurate and timely prediction of traffic flow condition can benefit both the traffic management agencies and individual drivers. For example, the real-time forecasting of average traffic speed on the highway can be utilized for the navigation service provider to plan routes and estimate the minimum time consumed by a vehicle from the source to the destination. Moreover, a better prediction may help the management agencies to make better travel decisions, therefore alleviate the terrible traffic congestion in cities, reduce the carbon-dioxide ($CO_2$) emissions, and improve the traffic operation efficiency.

With the development of traffic sensor technologies and widespread traffic monitoring system, the traffic data including speed, flow, and occupancy are exploding. The abundant real-time traffic data collected from various sources will provide us the basis for prediction. Undoubtedly, we have run into the era of big data transportation, the data-driven methods in traffic management and prediction are now becoming the most influential trend. The goal of the traffic prediction is to forecast the future traffic condition like speed in a sensor network given the historical traffic data and the spatial structure of the sensor network.

The forecasting task is challenging mainly because of the inherent complex spatial and temporal dependencies of the traffic data. On the one hand, the spatial dependencies in traffic are usually directional and non-Euclidean, for example two roads are close in Euclidean distance, they may behave very differently when they are in the opposite directions. It’s also generally accepted that the downstream traffic speed is more influential than the upstream one. On the other hand, the spatial
dependencies between traffic links may change over times, like the non-recurrent traffic conditions in the rush hours often cause non-stationarity and increase the correlation between the nearby links. The strong temporal dynamics in the traffic time series make it difficult to do long term prediction.

1.2 Motivation

Two factors have motivated us to undertake this study. The first one is, in previous research of our group, an energy efficient route and speed planning scheme was proposed [3], using accurate average traffic speed prediction, the planning would be more realistic. An accurate prediction can be also used for other research purposes. The second one is, with the availability of a huge amount of historical traffic data gathered sensors and new breakthroughs in data mining and deep learning, we want to explore the possibility of accurate predicting on a large road network.

1.3 Problem definition

The goal of traffic prediction is to predict the traffic speed in the future horizon given previously observed traffic flow from sensors on the road network. In most of the literature, methodologies trying to capture the stochastic nature of traffic flow are presented with details, however, the definition of traffic prediction problem is vague. For this reason, before introducing any modeling techniques, we first give a formal mathematical description of the traffic prediction problem.

Without loss of generality, neglecting all the gradient information of road segments or other unrelated conditions, the sensor network can be represented as a weighted directed graph \( G = (V, E, D) \), where \( V \) is a set of sensors distributed on the road segments, \(|V| = N\) is the number of sensors in the whole network; \( E \) is the set of edges such that a directed edge \( e_{ij} \) represents vehicles can move from sensor \( v_i \) to sensor \( v_j \) on the road network; \( D \in \mathbb{R}^{N \times N} \) is a weighted adjacency matrix (e.g., a function of the road network distance) representing the proximity of sensors. Let \( X_t \in \mathbb{R}^{N \times P} \) denotes the observed traffic signal of the sensor network at time \( t \), where \( P \) represents the number of features observed at each node, like speed, volume, etc. In this research, we only care about the average speed at each node in the network, i.e., \( X_t \in \mathbb{R}^{N} \).

Given the detected historical traffic speed data \([X_t, \ldots, X_{t-T}; G]\), the problem of traffic forecasting aims to find a function \( f(\cdot) \) which maps historical data to future
signals:

\[ [\hat{X}_{t+1}, \ldots, \hat{X}_{t+T}] = f([X_t, \ldots, X_{t-T'}; G]) \]

where \( T' \) denotes the amount of historical data we use, \( T \) is the prediction horizon. For the sake of simplicity, we first investigate the one-step prediction. One objective in finding the \( f(\cdot) \) is that the sum of squared error of the prediction is minimized:

\[
\min_{f \in F} \| X_{t+1} - f([X_t, \ldots, X_{t-T'}; G]) \|^2
\]

where \( F \) is a chosen family of prediction functions.

1.4 Objectives and Research Scope

The main objective of this research is to develop a prediction model for the short term traffic flow speed on a large road network. There is no intention of this project to design or evaluate as many models as in the literature. Specific objectives of this project are:

1. To review the state of the art prediction models in literature.
2. To alleviate the curse of dimensionality in the traffic prediction of a large road network.
3. To develop traffic flow prediction models based on existing statistical, and machine learning techniques.
4. To evaluate the performance of proposed models.

The research scope of this project is limited to predict traffic flow speed in the short term using the historical data and road network structure. Weather conditions, gradients of road segments, speed limits, or public events are not taken into consideration. "Short term", in this research, means we are concentrated on predictions within a very short time horizon, typically from several minutes to few hours. In addition, the traffic flow speed refers to as the average speed of vehicles on the road segment detected by sensors.

1.5 Thesis outline

In this section, we introduce the outline of this project.
• Chapter 1: Introduction - The research background, motivation, problem definition, and research objectives are presented.

• Chapter 2: Literature Review - We review the most influential studies on traffic flow prediction over the last ten years.

• Chapter 3: Methodology - The methodology of this research are provided in this chapter.

• Chapter 4: Experiment - We conduct experiments using the proposed methodology and several existing methods. Results and performance evaluation follows the experiment.

• Chapter 5: Conclusion - Conclusion and future works are presented.
Chapter 2

Literature review: traffic prediction approaches

Over the past few decades, a number of approaches for the traffic forecasting problem have been proposed. Generally speaking, there are two categories of the approaches: the knowledge-based methods in transportation and operational research which usually simulate the behavior of drivers in traffic, and the data-driven methods from the time series and data mining community. In this research, we focus mainly on the data-driven approaches. The various data-driven methods can be divided into four groups: the naive, parametric, non-parametric and hybrid approaches.

2.1 Naive approaches

In the naive approaches for traffic speed prediction, no parameters are required for calculation. They are simple, intuitive and often can be used as a baseline but without any research potential. The simplest method for short term traffic speed prediction would be just taking the latest observation, which is

\[ \hat{X}_{t+1} = X_t \]

A corresponding variant for highly seasonal time series data is

\[ \hat{X}_{t+1} = X_{t+1-T} \]

where \( T \) is the pre-specified period.

The historical average is another simple heuristic method for traffic speed pre-
prediction, which can be defined as

\[ \hat{X}_{t+1} = \frac{X_t + X_{t-1} + \ldots + X_{t-n}}{n} \]

where \( n \) is the number of chosen steps. Similarly, for the highly seasonal time series, the corresponding variant of historical average is

\[ \hat{X}_{t+1} = \frac{(X_{t+1-T} + X_{t+1-2T} + \ldots + X_{t+1-nT})}{n} \]

The naive approaches can be used for the highly self-correlated seasonal traffic pattern. However, for the complex road network, the parametric or sophisticated non-parametric approaches are widely favored.

\section{2.2 Parametric approaches}

Some parametric models including linear regression, ARIMA, and Kalman filter have been applied to the traffic prediction problem. The main characteristic of parametric models is that the number of parameters is fixed, we need to estimate the values of the parameters. The parametric models often perform quite well even without a large amount of data. Various statistical tests can also be used to evaluate the performance of the parametric models.

\subsection{2.2.1 Linear regression}

The most fundamental parametric model is the linear regression, which expresses the response variable \( y \) as a linear combination of predictor variables \( x_1, x_2, \ldots, x_n \). The general formulation of linear regression is

\[ y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \ldots + \beta_n x_{ni} + \epsilon_i \]

where \( \beta_0, \beta_1, \beta_2, \ldots, \beta_n \) are regression coefficients; the random error \( \epsilon_i \) is often assumed independently and identically normally distributed. For the matrix form, the linear regression can also be written as

\[ y = X \beta + \epsilon \]

The values of regression coefficients can be estimated using Ordinary Least Square or other classical methods, and are often given as

\[ \hat{\beta} = (X^T X)^{-1} X^T y \]
where \( \hat{\beta} \) stands for the estimated coefficients.

The ordinary linear regression model neglects the impact of road network topology; to model the varying relationships among sensors, the Geographically Weighted Regression (GWR) model is proposed in [4]. The GWR model is formulated as:

\[
y_i = \beta_0(u_i, v_i) + \sum_{i=1}^{n} \beta_i(u_i, v_i)x_i + \epsilon_i
\]

where \( \beta_i(u_i, v_i) \) represents the space-specific coefficients for predictor \( x_i \) measured on at geographic coordinates of \( (u_i, v_i) \). The corresponding estimator is given by

\[
\hat{\beta}(u_i, v_i) = (X^TW(u_i, v_i)X)^{-1}X^TW(u_i, v_i)y
\]

where \( W(u_i, v_i) \) represents a matrix of geographic weights specific to each location \( (u_i, v_i) \).

### 2.2.2 ARIMA

The autoregressive integrated moving average (ARIMA) model is a popular class of parametric models in the community of time series. Although the ARIMA model often requires the stationarity of time series, it has been very successful in short term traffic prediction. The ARIMA model, in some sense, can be seen as an extension of tradition linear regression model, which is constituted of two basic components - AR (autoregressive) and MA (moving average).

Just like the linear regression, where the predictors are the past \( p \) step values, the AR model can be expressed as

\[
x_t = \phi_1x_{t-1} + \phi_2x_{t-2} + \ldots + \phi_px_{t-p} + \epsilon_t
\]

where \( \phi_1, \phi_2, \ldots, \phi_p \) are the regression coefficients that need to be estimated; similarly, \( \epsilon_t \) are assumed to be independent, identically distributed. While in the MA model, the the predictors are the past \( q \) step disturbances,

\[
x_t = \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2} + \ldots + \theta_q\epsilon_{t-q} + \epsilon_t
\]

where \( \theta_1, \theta_2, \ldots, \theta_q \) are the parameters to be chosen. Combining the AR and MA model together, we get the ARMA model

\[
x_t = \phi_1x_{t-1} + \phi_2x_{t-2} + \ldots + \phi_px_{t-p} + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2} + \ldots + \theta_q\epsilon_{t-q} + \epsilon_t
\]
Using the backshift operator \( B \), where \( B^d = x_{t-d} \), we can rewrite the ARMA model as

\[
\phi(B)x_t = \theta(B)\epsilon_t
\]

where

\[
\phi(z) = 1 - \phi_1 z - ... - \phi_p z^p \\
\theta(z) = 1 + \theta_1 z + ... + \theta_q z^q
\]

Typically, differences will be utilized to decompose the trend and seasonality for some non-stationary data.

As early as 1970s, the ARIMA model has been used for short term traffic flow forecasting. In 2003, based on the Wold decomposition theorem and the assumption that a one-week lagged seasonal difference applied to traffic condition data will yield a weakly stationary transformation, in [3], the authors presented a theoretical foundation for modeling univariate traffic condition data streams as seasonal autoregressive integrated moving average (SARIMA) processes. Experimental analysis of two representative data sets, M25 Motorway and Interstate 75, showed that their three parameter SARIMA \((1, 0, 1)(0, 1, 1)s\) predictions consistently outperformed heuristic forecast benchmarks. After that, [6] implemented a dynamic SARIMA model for short-term traffic flow forecasting.

The univariate ARIMA model omits the possible spatial correlation. For the multiple time series data, a natural extension of ARIMA model is the Space-Time ARIMA (STARIMA) model [7]. Assume \( X_t \) is the \( N \times 1 \) vector of observations at time \( t \) at the \( N \) locations within the road network, the seasonal STARIMA model family is expressed as,

\[
\Phi_{P,\Lambda}(B^S)\phi_{p,\lambda}(B)\nabla_s^D \nabla^d x_t = \Theta_{Q,M}(B^S)\theta_{q,m}(B)\epsilon_t
\]

where

\[
\Phi_{P,\Lambda}(B^s) = I - \sum_{k=1}^{P} \sum_{l=0}^{\Lambda_k} \Phi_{kl} W_i B^k \nabla_s, \quad \phi_{p,\lambda}(B) = I - \sum_{k=1}^{p} \sum_{l=0}^{\lambda_k} \phi_{kl} W_i B^k, \\
\Theta_{Q,M}(B^s) = I + \sum_{k=1}^{Q} \sum_{l=0}^{M_k} \Theta_{kl} W_i B^k \nabla_s, \quad \theta_{q,m}(B) = I + \sum_{k=1}^{q} \sum_{l=0}^{m_k} \theta_{kl} W_i B^k.
\]

In the formulation above, \( \Phi_{kl} \) and \( \phi_{kl} \) are the seasonal and nonseasonal autoregressive parameters with temporal lag \( k \) and spatial lag \( l \), respectively; similarly, \( \Theta_{kl} \) and \( \Theta_{kl} \) are the seasonal and nonseasonal moving average parameters. \( P \) and \( p \) are the
orders for the seasonal and nonseasonal autoregression, $Q$ and $q$ are the seasonal and nonseasonal moving average orders. $\Lambda_k$, $\lambda_k$ are the seasonal and nonseasonal spatial orders for the $k^{th}$ autoregressive term, $M_k$ and $m_k$ are the seasonal and nonseasonal spatial orders for the moving average term. $\nabla^D_s$ and $\nabla^d$ are the seasonal and nonseasonal difference operators, $D$ and $d$ are, respectively, the number of seasonal and nonseasonal differences required. The random term, $\epsilon_t$ satisfies:

$$E[\epsilon_t] = 0, \quad E[Z_t \epsilon_{t+s}] = 0 \text{ for } s > 0,$$

and

$$E[\epsilon_t \epsilon_{t+s}] = \begin{cases} \sigma^2, & \text{if } s = 0, \\ 0, & \text{otherwise.} \end{cases}$$

$W_l$, a square $N \times N$ matrix, is the $l^{th}$ order weight matrix where the elements $w_{ij}^{(l)}$ is non-zero only if locations $i$ and $j$ are “$l^{th}$ order neighbors”, in this paper, $i$ and $j$ are $l^{th}$ order neighbors they are $l$–time reachable. The weights are taken $w_{ij}^{(l)}$ so that $\sum_{i=1}^N w_{ij}^{(l)} = 1$. Since every sensor isn the zero-th order neighbour of itself, $W_0$ is chosen as the identity matrix. If there is no seasonal component, the seasonal STARIMA model collapses to the form (STARIMA)

$$Z_t = \sum_{k=1}^{p} \sum_{l=0}^{\lambda_k} \phi_{kl} W_l Z_{t-k} + \sum_{k=1}^{q} \sum_{l=0}^{m_k} \theta_{kl} W_l \epsilon_{t-k} + \epsilon_t.$$

STARMA models can be viewed as special cases of the Vector Autoregressive Moving Average (VARMA) models.

As a special case of the Vector Autoregressive Moving Average (VARMA) model, STARIMA method provides a great reduction in the number of parameters. In STARIMA model, the spatial topological relationships of a road network are captured through a hierarchical ordering weight matrices for the neighbors. The elements of the $l^{th}$ order weight matrix are nonzero only in the case that the locations $i$ and $j$ are “$l^{th}$ order neighbors”. This implies that in the formulation of STARIMA model, the autoregressive parameters are nonzero only if they are $l^{th}$ order correlated. However, the construction of the order weight matrix is sometimes tricky.

### 2.2.3 Kalman filter

Another parametric technique widely used in traffic prediction problem is the Kalman filters proposed by Kalman in [8]. The authors of [9] proposed two models incorporating the Kalman filtering theory to predict the short-term traffic conditions. The major advantage of the two models is that they utilized the estimated future data
to update the error for better prediction. Testing results indicated that they are robust for long-term prediction. In order to reduce local noises in the short-term traffic data and improve prediction accuracy, the discrete wavelet decomposition technique was used to divide the original data into several approximate and detailed data, then the Kalman filter model was applied [10]. The authors showed that the wavelet Kalman filter model outperformed the direct Kalman filter model. Other approaches employing the Kalman filter techniques can also be seen in [11] and [12].

2.3 Nonparametric approaches

Parametric methods are appreciated for their exact formulation and possible statistical meaning. However, they usually rely on the assumption of stationarity and linear correlations of the time series. These assumptions are often violated in the traffic data. On the contrary, the non-parametric methods like K Nearest Neighbors (K-NN), Support Vector Machines (SVM) and Neural Networks (NN) perform significantly better than the parametric methods when modeling the complex nonlinear data.

2.3.1 K nearest neighbors

The K-nearest neighbour approach in short term traffic prediction is favored for its simplicity in the model formulation of multivariate data, independence of the assumption on the traffic conditions and intuitive explanation [13]. [14] may be the first one that suggested the K-NN approach as a candidate forecaster which may sidestep the problems inherent in parametric approaches. Whereas, the empirical study revealed that their K-NN method performed comparably to, but not better than, the linear time-series approach. A possible explanation is the lack of data, since the authors used only about one and half hour of data in their experiments. [15] and [16] further demonstrated the performance of K-NN algorithms. Nevertheless, for the K-NN method, the distance measure and value of K are disputable in application.

2.3.2 Support vector machine

Because of the great generalization ability and guarantee of global minima for given training data, Support Vector Machines (SVM) have been widely used in the classification and regression problems. The basic idea behind SVM is to find a hyperplane to classify the data. To address the linearly non-separable problems, we can map the input data into a feature space where the data is linearly separable. We can also use support vector regression (SVR) to solve the regression problem. Generally,
suppose the training dataset is $D = \{(x_i, y_i)\}_{i=1}^{n}$. The goal of SVM is to find the optimal hyperplane such that the relationship between $x_i$ and $y_i$ is like

$$f(x_i) = w^T \phi(x_i) + b$$

where $\phi$ is a non-linear mapping from the input data space to a feature space. To train SVR, we need to do the following optimization problem:

$$\min \frac{1}{2} w^T w + C \sum (\xi_i + \xi_i^*)$$

s.t. $y_i - f(x_i) \leq \epsilon + \xi_i$,

$f(x_i) - y_i \leq \epsilon + \xi_i^*$,

$\xi_i, \xi_i^* \geq 0$

where $\xi_i$ and $\xi_i^*$ are slack variables, $\epsilon$ and $C$ need to be predetermined before training.

The authors of [17] applied the support vector regression (SVR) for travel-time prediction. Their experimental results of travel-time prediction over a short distance in rush hour reflected the traffic patterns that are quite different from the past average. They said that their SVR predictor significantly outperformed the Current-time predictor and Historical-mean predictor. However, to fully demonstrate the efficiency of their approach, a comparison of their method with STARIMA or Kalman filter models is needed. To predict short-term traffic flow under atypical conditions, such as vehicular crashes, inclement weather, a supervised statistical learning technique called Online Support Vector machine for Regression (OL-SVR) was applied in [18]. They stated that compared with the three well-known prediction models including Gaussian maximum likelihood (GML), Holt exponential smoothing, and artificial neural net models, the OL-SVR model is the best performer under non-recurring atypical traffic conditions.
2.3.3 Gaussian regression

In the traffic prediction, the Gaussian processes regression (GPR), a kernel-based learning algorithm like SVM, is another data-driven solution with a big data potential. In GPR, the time series of traffic speed are modeled as a Gaussian Processes. \( X_t \), the traffic speed can be modeled as:

\[
X_t = f(t) + \epsilon_t
\]

where \( f(t) \) is a Gaussian Process and \( \epsilon_t \) is observation noise following an independent, identically distributed Gaussian distribution with zero mean and variance \( \sigma^2_t \), i.e., \( \epsilon_t \sim N(0, \sigma^2_t) \). The key point of using GPR is to design an appropriate kernel function which can reflect the characteristics of the historical data. Once the kernel function is designed, we can use MLE estimation and Gradient Descent algorithm to learn the parameters in the covariance function.

To take various traffic behaviors such as periodicity and self-similarity into account, in [19], the Gaussian process regression was adapted in traffic modeling and prediction. A Hurst estimation method built on machine learning techniques was exerted to connect the traffic characteristic and GPR parameters. A method called vicinity Gaussian Processes in [20] was proposed to provide a flexible framework for traffic prediction in the context of missing data and other measurement errors in the vehicular traffic network. They derived a dissimilarity matrix on the weighted directed graph of the network, which accounted for the selection of training subsets. Experimental results showed that the root mean square error of prediction by the vicinity Gaussian Processes method reached 18.9% average improvement when the training subsets were selected appropriately. However, it’s debatable in which case the training subsets are appropriate. Comparison between the vicinity Gaussian Processes with other methods is also critical to evaluate the efficiency of vicinity Gaussian Processes. Based on historical data collected in Dublin city, the authors of [21] first used a discrete time Gauss-Markov model to predict future traffic saturations at junctions of the street with sensors. Then a Gaussian Process derived from the street graph to extend these predictions to junctions without sensors.

2.3.4 Bayesian network

Bayesian networks (BNs), also known as belief networks, is a kind of probabilistic graphical model (GMs) [22]. Corresponding to the directed acyclic graph (DAG), another GM structure, BNs is popular in the statistics, machine learning, and artificial intelligence societies. Formally, a Bayesian network can be defined as a pair
\((G, P)\), where \(G\) is a DAG constituted of nodes \(X\), \(P = p(x_1|\pi_1), ..., p(x_n|\pi_n)\) is a set of conditional probabilities with \(\pi_i\) is the set of parent nodes of node \(x\). The graphical structures of \(G\) represent knowledge in the uncertain domain. Particularly, the nodes of the graph denote random variables, while the edges between the nodes represent direct causal dependencies among the corresponding random variables. The joint probability \(p(X)\) is formulated as

\[
p(X) = \prod_{i=1}^{n} p(x_i|\pi_i)
\]

For the Gaussian Bayesian network, the joint probability distribution is defined explicitly as

\[
f(x) = (2\pi)^{-n/2}\Sigma^{-1/2}\exp\{-1/2(x - \mu)^T\Sigma(x - \mu)\}
\]

which is the density function of the multivariate normal distribution \(N(\mu, \Sigma)\). One advantage of the Bayesian network is that it can be used very easily to model the multivariate traffic flow data.

In [23], following the intuitive causal relationship, the authors modeled the traffic flows among adjacent road links in a transportation network as a BN. The joint probability distribution is described as a Gaussian mixture model (GMM), where the parameters are computed with the competitive expectation maximization (CEM) algorithm. They found the performance of the Bayesian network is significantly better than the ordinary AR method. To model the non-stationary characteristics of traffic flows, the authors of [24] proposed an adaptive Bayesian network where the network topology may change over phases of traffic flows. With a statistical analysis of real traffic data, they claimed that the graph topology can be adapted to the local traffic phase. One can refer [25], [26], [27] for other approaches utilizing the Bayesian network model.

### 2.3.5 Neural Network

To imitate the human brain, Artificial Neural Networks (ANN) were designed in 1940s [28], [29]. Yet they have been hugely successful in dealing with a number of difficult tasks, especially recently. ANNs’ capabilities make them potentially valuable for situations: (1) large data sets; (2) with nonlinear structure; (3) the multivariate time series forecasting problems [30], [31]. The flexible structure of neural networks and various convolution operations constitute plenty of short-term traffic flow prediction models. In this research, we give a glimpse of some typical neural network models.
A Back-Propagation neural network were trained to make short-term forecasts of traffic flow, speed, and occupancy in [32]. Even though not out-performing the naive predictors, the empirical results for occupancy and flow forecasts showed some promise. [33] developed a time-lag recurrent network (TLRN) to predict short-term traffic conditions. The experimental results indicate that the method is capable of predicting the short-term future speed with a high degree of accuracy. Most recently, the deep learning methods have been developed for traffic forecasting in [34] and [35]. A novel deep architecture combined CNN and LSTM was introduced in [36]. They exploited a 1-dimension CNN to capture spatial features, and two LSTMs to mine the short-term variability and periodicities of traffic flow.

To incorporate the spatial-temporal dependency in the traffic flow, a deep learning framework for traffic forecasting, Diffusion Convolutional Recurrent Neural Network (DCRNN), was proposed in [1]. The spatial dependency of traffic flow was captured through bidirectional random walks on the graph, while the temporal dependency was captured the encoder-decoder architecture with scheduled sampling. In result analysis of their paper, the proposed approach obtained significantly better performance than baselines when evaluated on two real-world traffic datasets. The number of sensors in their traffic datasets are 207, 325 respectively.

Figure 2.2: System architecture for the DCRNN designed for spatiotemporal traffic forecasting. The historical time series are fed into an encoder whose final states are used to initialize the decoder. The decoder makes predictions based on either previous ground truth or the model output [1].

Replacing the DCRNN with more powerful Graph Attention LSTM Network (GAT-LSTM), a novel neural network architecture which can operate on graph-structured data, the authors of [37] constructed an end-to-end trainable encoder-forecaster model to solve traffic flow forecasting problem on graphs. Since the public multi-link traffic flow data are scarce, the author built such a dataset in the road.
network of Guiyang that contains 112 intersections. Through experiments, they showed their GAT-LSTM model has achieved state-of-the-art results. Under the consideration of uncertain traffic accident factors, a novel fuzzy-based convolutional neural network (F-CNN) method was proposed in [38]. The key idea in their paper is to introduce fuzzy representation into the deep learning model therefore to lessen the impact of data uncertainty. In their experiment, the historical traffic flow data was given on $32 \times 32$ grid regions.

To address the challenges in modeling dynamic spatiotemporal dependencies among network-wide links and long-term traffic prediction for the next few hours, a spatiotemporal attentive neural network (STANN) for the network-wide and long-term traffic prediction was provided in [2]. Like in other papers, the encoder-decoder architecture is also utilized in STANN with the attention mechanisms. The authors conducted experiments over three different traffic datasets in Hong Kong, where there are 605 links real-time traffic speeds in total. One limitation of their work is that the dimension of the spatial attention vector needs to be very large as the network size is large.

![Figure 2.3: The architecture of STANN with two components: the encoder for modeling spatio-temporal dependencies and the decoder for multi-step traffic prediction](image)

2.4 Hybrid approaches

To enhance the prediction accuracy, many hybrid methods in literature have been tried in recent years. The ATHENA model employed in [39] may be the first hybrid
approach used for short term traffic prediction. In the ATHENA model, the traffic data was grouped by a clustering method, then for each cluster, the traditional linear regression model was applied. In another hybrid method known as KARIMA [40], the Kohonen self-organizing map was used as an initial classifier, then an individually tuned ARIMA model was applied for each class. The authors pointed out that their KARIMA method outperformed the straightforward ARIMA model or the ATHENA model on a French motorway.

The author of [41] developed two hybrid approaches where the Self-Organising Map (SOM) was employed to classify the traffic of the road network into different states. The first hybrid approach included four ARIMA model, while the second one used two Multi-Layer Perception (MLP) models. In addition to the superior forecasting performance of proposed models, they also analyzed the effects of different proportions of missing data. Autoregressive Integrated Moving Average with Generalized Autoregressive Conditional Heteroscedasticity (ARIMA-GARCH) model was proposed in [42] for traffic flow prediction. It combined the popular linear ARIMA model and nonlinear GARCH model to create a non-linear hybrid prediction model. The preprocessed time series was first treated with the ARIMA model, the error series of ARIMA model was then fitted with the GARCH model. It was not surprising that the performance of the hybrid model was better compared with the standard ARIMA model. However, the author indicated that the introduction of conditional heteroscedasticity may be unnecessary since it didn’t bring satisfactory improvement in prediction accuracy. In some cases, the general GARCH(1,1) model may even deteriorate the performance.

Generally, the hybrid methods are better performed than the compared simple single models. However, the computation of hybrid methods is rather difficult, and often lack of intuitive explanation. Although the ANN, deep learning or hybrid approaches can handle the nonlinearity, nonstationarity of the dynamic traffic flow, the main disadvantage is that they often require a large number of training samples. This drawback will lead to a time-consuming training phase even if we have enough training data, therefore reduce the applicability of the predictors in real-time traffic prediction. Another disadvantage of the predictors is that most of them are analytically intractable.
Chapter 3

Methodology

In most of the literature listed above, the road network concerned is rather small, often consisted of several hundred sensors in the whole network. When it comes to a large road network with thousands of sensors, many of the methods mentioned above is complicated to implement. Because for high dimensional data sets, a large number of parameters are required to be estimated but not all of them are necessary. It may be a potential direction to cluster the time series at each sensor into different groups.

3.1 Time series clustering

Clustering is a data mining technique, a possible solution for classifying enormous data when we have no prior knowledge about classes [43]. It’s a practical approach to find possible hidden patterns or similarity in data. Nowadays, clustering has been applied on time series data generated by real-world applications to gain insight into the data. However, unlike the static data clustering built on the Euclidean distance, time series clustering requires a good distance measure for time series data. Therefore, we first review the distance measures occur the most in time series clustering literature [44].

3.1.1 Shape-based distances

Mikowski distance ($\forall p$) is a generalization of Euclidean distance. Let $X_i$ and $X_j$ each be a $n$-dimensional vector, the Mikowski distance ($\forall p$) is defined as

$$d_M(X_i, X_j) = \left( \sum_{k=1}^{n} |X_{ik} - X_{jk}|^p \right)^{\frac{1}{p}}$$
Shape-based distances

<table>
<thead>
<tr>
<th>Lock-step measures</th>
<th>Elastic measures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minkowski ($\forall p$)</td>
<td>Dynamic Time Warping (DTW)</td>
</tr>
<tr>
<td>Pearson correlation</td>
<td>Longest Common Subsequence (LCSS)</td>
</tr>
</tbody>
</table>

Feature-based distances

| Discrete Fourier Transform (DFT) | Discrete Wavelet Transform (DWT) |

Table 3.1: Most commonly used (and effective) distance measures

where $p$ is a positive integer. Manhattan distance ($p = 1$) and Chebyshev distance ($p = 1$) are special cases of Minkowski distance. The time complexity of computing the Minkowski distance ($\forall p$) is $O(n)$ and thus it takes $O(nN^2)$ time to determine the distance matrix with this measure for $N$ vectors.

To take the linear association of two vectors of variables, the Pearson correlation distance is defined by using the Person correlation coefficient, where the Person correlation coefficient is

$$
\rho(X_i, X_j) = \frac{\text{Cov}(X_i, X_j)}{\sigma_{X_i} \sigma_{X_j}} = \frac{\frac{1}{n} \sum_{k=1}^{n} (X_{ik} - \bar{X}_i)(X_{jk} - \bar{X}_j)}{\sqrt{\frac{1}{n} \sum_{k=1}^{n} (X_{ik} - \bar{X}_i)^2} \sqrt{\frac{1}{n} \sum_{k=1}^{n} (X_{jk} - \bar{X}_j)^2}}
$$

where $\mu_{X_i}$ and $\mu_{X_j}$ are the means of $X_i$ and $X_j$, $\sigma_{X_i}$ and $\sigma_{X_j}$ are the standard deviations of $X_i$ and $X_j$, respectively. Note that the value of $\rho$ lies within $[-1, 1]$, and it is invariant for scaling. Then, the Pearson correlation distance is defined as

$$
d_{\text{cor}}(X_i, X_j) = 1 - \rho(X_i, X_j)
$$

The time complexity for computing the Pearson correlation distance is the same as Minkowski distance. Alternative correlation distance measures use Spearman’s Rank or Kendall’s Tau correlation coefficients which indicate correlation based on rank and are less sensitive to noise and outliers compared to the Pearson correlation coefficient [45]. However, the time complexities of these two distance are much larger, $O(n\log n)$ for Spearman’s Rank and $O(n^2)$ for Kendall’s Tau [46].

Dynamic Time Warping (DTW) is a generalization of classical algorithms for comparing discrete sequences to sequences of continuous values. When computing DTW distance for two given sequences, $X = (x_1, ..., x_n)$ and $Y = (y_1, ..., y_n)$, first an
\((n \times m)\) local cost matrix (LCM) is calculated, which contains the distance \(d(x_i, x_j)\) between two points \(x_i\) and \(x_j\). For \(d(x_i, x_j)\), the Euclidean distance is normally used, like \(d(x_i, x_j) = (x_i - x_j)^2\). Next, a warping path, \(W = w_1, \ldots, w_K\) is determined, where \(\max(m, n) \leq K \leq m + n - 1\). The path is set of elements in LCM that satisfies three constraints: boundary condition, continuity, and monotonicity. The boundary condition requires the warping path to start and end in the diagonal corners of the LCM: \(w_1 = (1, 1), w_K = (n, m)\). The continuity constraint restricts the allowable steps to adjacent cells. The monotonicity constraint forces the points in the warping path to be monotonically spaced in time. The total distance for path \(W\) is obtained by summing the individual elements (distances) of the LCM that the path traverses. To obtain the DTW distance, the path with minimum total distance is required. This path can be obtained by an \(O(nm)\) algorithm that is based on dynamic programming (DP). The following DP recurrence can be used to find the path with minimum cumulative distance:

\[
d_{\text{cum}}(i, j) = d(x_i, y_j) + \min\{d_{\text{cum}}(i - 1, j), d_{\text{cum}}(i, j - 1), d_{\text{cum}}(i - 1, j - 1)\}
\]

We now obtain the DTW distance by summing the elements of the path with minimum cumulative distance [17],

\[
d_{\text{DTW}} = \min \sum_{k=1}^{K} w_k
\]

where \(w_k\) is the distance that corresponds to the \(k^{th}\) element of warping path \(W\).

Longest Common Subsequence (LCSS) similarity measure, just as its name implies, aims to find the longest subsequence that is common to two or more sequences. The LCSS distance for real number sequences can be obtained by using recursion:

\[
L(i, j) = \begin{cases} 
0, & \text{if } i = 0 \text{ or } j = 0, \\
1 + L(i - 1, j - 1), & \text{for } |x_i - y_j| < \epsilon, \\
\max\{L(i - 1, j), L(i, j - 1)\}, & \text{otherwise}
\end{cases}
\]

where \(1 \leq i \leq n\) and \(1 \leq j \leq m\). The distance of \(X\) and \(Y\) is now computed by solving \(L(n, m)\). The scaled version of LCSS is defined as

\[
d_{\text{LCSS}} = \frac{n + m - 2L(n, m)}{n + m}
\]

The time complexity for computing LCSS distance is \(O(nm)\).
3.1.2 Feature-based distances

Introduced by [48], the Discrete Fourier Transform is a dimensionality reduction method, which transforms the time series from a "time-domain" \( x(t) \) to a "frequency-domain" representation \( X(f) \). The DFT is defined as,

\[
X(l) = \sum_{k=0}^{n-1} x_k e^{-\frac{i2\pi lk}{n}}
\]

for \( x = x_0, \ldots, x_{n-1}, l = 0, \ldots, n - 1 \) and \( i^2 = -1 \). The collection of values of \( X(f) \) at frequencies \( f \) are called the spectrum of \( x(t) \). The inverse DFT is defined as,

\[
x_k = \sum_{k=0}^{n-1} X(l) e^{\frac{i2\pi lk}{n}}
\]

for \( x = x_0, \ldots, x_{n-1}, l = 0, \ldots, n - 1 \). The inverse DFT transforms the spectrum \( X(f) \) back to the "time-domain". As indicated in [48], most of the energy in real-world signals (time series) is concentrated in the low frequencies. The advantage of the DFT arises: it can be used to reduce the number of dimensions of a time series by only considering a limited number \( q(q \leq n) \) of frequencies. To avoid approximating a time series with too few frequencies, the first \( q = \frac{n}{2} \) frequencies should be used. Then the Euclidean distance between the first \( q \) frequencies of the DFT is calculated to approximate the Euclidean distance between the original time series. The time complexity of DFT is \( O(n^2) \), while calculating the distance between two time series based on the Fourier coefficients has time complexity \( O(q) \). Thus the whole process of determining the Fourier distance has time complexity \( O(n^2) \).

The Discrete Wavelet Transform (DWT), just like the DFT, is another dimensionality reduction method that also reduces noise. It decomposes a time series into a set of basis functions that are called wavelets. Wavelets contain two function: the wavelet function \( \psi \) and the scaling function \( \varphi \), which are also referred to as the mother wavelet and father wavelet, respectively. The simplest kind of wavelet function, the Haar wavelet is introduced in [49], it’s defined as

\[
\psi(t) = \begin{cases} 
1, & \text{if } 0 < t \leq \frac{1}{2}, \\
-1, & \frac{1}{2} < t \leq 1, \\
0, & \text{otherwise}.
\end{cases}
\]
\[ \varphi(t) = \begin{cases} 1, & \text{if } 0 < t \leq 1, \\ 0, & \text{otherwise}. \end{cases} \]

Using the Pyramid algorithm in [50], the time series can be decomposed into different levels of approximation and detail coefficients. Like the DFT, distance measure based on DWT is calculated using the Euclidean distance of decomposed coefficients. The advantage of DWT is that it also captures information on the location in time.

### 3.1.3 Clustering methods

There has been various types of clustering method, in this paper, we only concentrated on two approaches, the k-means and k-medoids. Given a set of observations \((x_1, x_2, ..., x_n)\), k-means is a clustering method that aims to partition the \(n\) observations into \(k(k \leq n)\) clusters, \(S = \{S_1, S_2, ..., S_k\}\). Each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. Formally, the objective is to find,

\[
\arg \min_S \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2 = \arg \min_S \sum_{i=1}^{k} |S_i| \text{Var}(S_i)
\]

Even though it was first proposed in 1955, it is still one of the most commonly used clustering methods. The most common algorithm, also referred to as Lloyd’s algorithm in the computer science community, uses an iterative refinement technique. Although it’s ubiquitous, the algorithm does not guarantee to find the optimum.

The k-medoids is a clustering algorithm reminiscent to the k-means algorithm. In contrast to the k-means algorithm, k-medoids chooses data points as centers (medoids or exemplars) and can be used with arbitrary distances, while in k-means the centre of a clusters is not necessarily one of the input data points. The most popular heuristic for k-medoids is the Partitioning Around Medoids (PAM) algorithm proposed in 1987.

### 3.2 Prediction models

#### 3.2.1 Clustering

Even though the deep neural network or other machine learning methods have gained extensive attention in the area of traffic prediction, when the road network is large,
say more than $8 \times 10^4$ sensors in our experiment, the architecture of NN is far beyond controllable. Rather than just implementing the NN to our case, we first propose to cluster the spatial time series into different groups. As mentioned above, a good distance measure is the most critical part of time series clustering. In this paper, to capture the geographical and temporal correlation of the traffic speed data, we employ the weighted combination of DWT and Euclidean distance, i.e.

$$d_{DWTE}(x, y) = w_1d_{DWT}(x, y) + w_2d_E(x, y)$$

where $w_1 \geq 0$, $w_2 \geq 0$, $w_1 + w_2 = 1$. When $w_1 = 0$, only the spatial correlation is considered in the clustering. For simplicity, the clustering algorithm we use is the standard k-means, i.e. Lloyd’s algorithm. The details of Lloyd’s algorithm can be found in Algorithm 1.

**Algorithm 1** k-means clustering algorithm (Forgy/Lloyd).

1: Decide on a value $k$.
2: Initialize $k$ cluster centers $\mu_1, \ldots, \mu_k$.
3: while stopping condition not met do
4:     for $i = 1 : N$ do
5:         $c_i := \arg \min_i d(x_i, \mu_i)$
6:     end for
7:     for $j = 1 : k$ do
8:         $\mu_j := \frac{\sum_{i=1}^{N} 1\{c_i=j\} x_i}{\sum_{i=1}^{N} 1\{c_i=j\}}$
9:     end for
10: end while
11: return $c_1, \ldots, c_N$ and $\mu_1, \ldots, \mu_k$.

After partitioning the whole network into several closely related regions, this research employs Historical Average, Gaussian Processes Regression (GPR), univariate ARIMA, Vector ARIMA (VARIMA), Stacked Autoencoders (SAEs) and Long Short-Term Memory (LSTM) Recurrent Neural Network (RNN) to examine the spatial and temporal dependency in traffic flow. In the following sections. Most of models have been introduced before, here we give more details about SAEs and LSTM RNN.
3.2.2 Stacked Autoencoders

A simple structure of the autoencoder is present in Figure 3.1, which has one input layer, one hidden layer, and one output layer. Given the training samples \( \{x_1, x_2, ..., x_T\} \), where \( x_t \in \mathbb{R}^d \), the autoencoder first encodes the input \( x_t \) to a hidden representation \( y(x_t) \), then decodes \( y(x_t) \) to the reconstruction \( z(x_t) \)

\[
y(x_t) = \sigma_1(W_1x_t + b_1), \\
z(x_t) = \sigma_2(W_2y(x_t) + b_2)
\]

where \( W_1, W_2 \) are the weighted matrices; \( b_1, b_2 \) are the bias vectors; \( \sigma_1, \sigma_2 \) are the activation functions, usually the sigmoid function. The model parameters are obtained by minimizing the reconstruction error \( L(X,Z) \), which is

\[
\theta = \arg \min_{\theta} L(X,Z) = \frac{1}{2} \sum_{t=1}^{N} \|x_t - z_t\|^2
\]

in which \( \theta \) denotes all the parameters involved.

A stacked autoencoders (SAEs) model is created by stacking the autoencoders hierarchically to form a deep network [51]. To be specific, considering a SAEs model contains \( l \) layers, the output of the \( k \)th hidden layer will be used as the input of the \( (k+1) \)th hidden layer, where \( 1 < k < l \); while the training set is the input of the first layer. A logistic regression layer needs to be added on the top of a SAEs network for traffic flow prediction. The network structure of SAEs is illustrated in Figure 3.2. Rather than using the gradient-based optimization back propagation method, We adopt the training algorithm in [51].
3.2.3 LSTM Recurrent Neural Network

Recurrent Neural Networks (RNN) is one state-of-art feedforward network for sequential data. In the feedforward neural network, the information only moves in one direction, from the input layer, through the hidden layers, to the output layer. The characteristic of RNN is that the input to the hidden layers consists of not only the external input but also activations from the previous step. Given a sequence $x = x_1, ..., x_T$, where $x_t \in \mathbb{R}^d$, the hidden state at time step $t$ of RNN will be calculated as

$$h_t^j = \begin{cases} 
0, & \text{if } t = 0, \\
\phi(h_{t-1}^j, x_t), & t > 0.
\end{cases}$$

where $x_t$ is current input for the hidden unit $j$, $h_{t-1}^j$ is the previous hidden state, $\phi$ is the activation function, such as a sigmoid function. The unfold structure of RNN can be seen in Figure 3.3.

Long Short-Term Memory (LSTM) Recurrent Neural Network (RNN) is a neural network architecture containing LSTM units which can remember values for either long or short time periods. Typically, LSTM units are contained in blocks as shown in Figure 3.4. A block contains several gates that control the information flow. In
a peephole, for example, the data flow is as follows:

\[
\begin{align*}
    f_t &= \sigma_g(W_f x_t + U_f c_{t-1} + b_f) \\
    i_t &= \sigma_g(W_i x_t + U_i c_{t-1} + b_i) \\
    o_t &= \sigma_g(W_o x_t + U_o c_{t-1} + b_o) \\
    c_t &= f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c c_{t-1} + b_c) \\
    h_t &= o_t \circ \sigma_h(c_t)
\end{align*}
\]

where \(x_t \in \mathbb{R}^d\) is the input vector of the LSTM unit, \(f_t, i_t, o_t \in \mathbb{R}^h\) are the activation of forget gate, input gate and output gate, respectively, \(h_t \in \mathbb{R}^h\) is the output vector of the LSTM unit, \(c_t \in \mathbb{R}^h\) is the cell state vector, \(W \in \mathbb{R}^{h \times d}, U^{h \times h}\) and \(b \in \mathbb{R}^h\) are the weight matrices and bias vector parameters which need to be learned during training. Moreover, \(\sigma_g\) is the sigmoid activation function for gate, \(\sigma_c\) and \(\sigma_h\) are the hyperbolic tangent activation function for the input and output of the block, respectively and \(\circ\) is the symbol of element-wise product.
Chapter 4

Experiment

4.1 Data set

To evaluate the performance of proposed models, the transportation network we choose follows [3], which contains the U.S. National Highway Systems (NHS) from the dataset of Clinched Highway Mapping (CHM) Project. The whole highway network consists of 84504 nodes (waypoints) and 89119 (one-direction) edges. We have collected the speed data of the network from the API provided by HERE map for about two weeks. The time interval of our data is 40 minutes. We use ten days of the data as the training set and four days of the data as the test set.

![U.S. National Highway Network](image)

Figure 4.1: U.S. National Highway Network [3]

A sample of traffic flow speed on one edge is presented in Figure 4.2. From Figure 4.2, we can see a clear daily pattern of the traffic flow speed. Instead of putting all the raw data into a prediction model like in most of the literature, here we first using the historical average to eliminate the daily trend. Then, we put the residuals into the prediction models. After that, the predicted traffic flow speed is
recovered by adding the daily trend to the predicted residuals. An illustration of the procedure of the prediction is presented in Figure 4.3, where $z_{t-i}^j$ is the residual value of sensor $j$ at time $t - i$, $1 \leq j \leq m$, $1 \leq i \leq T$, $m$ is the number of sensors, $T$ is the historical horizon.

A sample of autocorrelation function (ACF) and partial autocorrelation function (PACF) for the residuals of speed data at one selected edge is presented in Figure 4.4. The residuals on other edges have a similar pattern. ACF and PACF of the residuals can be used as a guide when choosing the orders of the ARIMA model or determining the scale of input data for the neural network.
4.2 Evaluation metrics

In this section, we introduce the evaluation metrics appeared in literature. Suppose 
\( x = x_1, x_2, ..., x_n \) represents the ground truth, while \( \hat{x} = \hat{x}_1, \hat{x}_2, ..., \hat{x}_n \) denotes the predicted values, and \( \Omega \) stands for the indices of observed samples, the most common evaluation metrics in the literature are defined as follows,

Root Mean Square Error (RMSE)

\[
RMSE(x, \hat{x}) = \sqrt{\frac{1}{|\Omega|} \sum_{i \in \Omega} (x_i - \hat{x}_i)^2}
\]

Mean Absolute Error (MAE)

\[
MAE(x, \hat{x}) = \frac{1}{|\Omega|} \sum_{i \in \Omega} |(x_i - \hat{x}_i)|
\]

Mean Absolute Percentage Error (MAPE)

\[
MAPE(x, \hat{x}) = \frac{1}{|\Omega|} \sum_{i \in \Omega} \left| \frac{x_i - \hat{x}_i}{x_i} \right|
\]

4.3 Result

For the ARIMA and VARIMA model, grid search is applied to find the suitable orders, where we set maximum \( p = 7, \ d = 2, \ q = 7 \) for each edge. We choose the order which has the smallest value of Akaike information criterion (AIC). The Gaussian kernel is utilized for SVR and GPR. The evaluation metrics for the different algorithms we choose can be seen Table ?? . As shown in the table, for the one-step traffic flow speed prediction problem, the LSTM network with multivariate input has the overall best performance in terms of MAPE. Another neural network, the SAEs
network has a similar result. The experiment implies the deep neural network is suitable for capturing the spatial-temporal dependencies in the traffic flow speed. A sample prediction for four random selected road segments can be found in Figure 4.5.

<table>
<thead>
<tr>
<th>Prediction Algorithm</th>
<th>HA</th>
<th>ARIMA</th>
<th>VARIMA</th>
<th>SVR</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSE</td>
<td>4.62</td>
<td>4.33</td>
<td>2.94</td>
<td>3.72</td>
</tr>
<tr>
<td>MAE</td>
<td>3.50</td>
<td>3.37</td>
<td>2.34</td>
<td>2.76</td>
</tr>
<tr>
<td>MAPE</td>
<td>10.8%</td>
<td>9.4%</td>
<td>7.2%</td>
<td>8.7%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prediction Algorithm</th>
<th>GPR</th>
<th>SAEs</th>
<th>LSTM</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSE</td>
<td>3.97</td>
<td>2.80</td>
<td>2.75</td>
</tr>
<tr>
<td>MAE</td>
<td>2.95</td>
<td>2.32</td>
<td>2.20</td>
</tr>
<tr>
<td>MAPE</td>
<td>9.1%</td>
<td>5.8%</td>
<td>5.3%</td>
</tr>
</tbody>
</table>

In fact, as we can see in the Table 4.1, the parametric models like VARIMA have also achieved acceptable accuracy. It’s conceivable that in the highway network, the nonlinear dependencies between road segments is relatively weak than in urban road network. Compared with the urban traffic flow speed, the highway traffic speed seems more stable in most of the roads. We can see that the predict errors in this project is much smaller than the ones in Table 4.2.

<table>
<thead>
<tr>
<th></th>
<th>30 min</th>
<th></th>
<th>1 hour</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
<td>RMSE</td>
<td>MAPE</td>
<td>MAE</td>
</tr>
<tr>
<td>DCRNN</td>
<td>3.15</td>
<td>6.45</td>
<td>8.8%</td>
<td>3.60</td>
</tr>
<tr>
<td>GCRNN</td>
<td>3.24</td>
<td>6.74</td>
<td>9.0%</td>
<td>3.81</td>
</tr>
</tbody>
</table>

Table 4.2: Performance comparison for DCRNN and GCRNN on the METRA-LA dataset.

Interestingly, as we noticed in this research, compared with the complex neural network, the simple algorithm like the historical average or ARIMA even has better performance on some edges. A possible reason is that in the training process, the algorithm doesn’t converge to the optimal parameters. As far as I am concerned, when taking the uncertainty of training into account, it’s still disputable whether the nonparametric methods will outperform the advanced statistical approaches such as the VARIMA. In some situations, for simplicity and theoretical tractability, we can just use the parametric model to capture the dynamics of traffic flow. Such as when there’s a nonrecurrent accident happened on a road segment, we can control the related roads in the parametric model to alleviate the corresponding traffic jam. Further, there are many complex models in the literature, however, considering the
inherent randomness behind the traffic flow, there’s no much room for improving the prediction performance. Rather than focusing on utilizing different models to do the traffic prediction, investigating the impact of missing value would be another interesting direction.
Chapter 5

Conclusion

5.1 Conclusion

The main objective of this project is to find suitable approaches for the traffic flow speed prediction on a large highway system, which captures both the spatial and temporal dependencies of the traffic speed time series. Due to the accumulation of traffic flow data and advancement in computational power, the data-driven approaches have been attracting scientific focus for traffic flow prediction problem in the last decades. Therefore, in this project, we first review various state-of-art prediction models for short traffic flow prediction problem. In the literature, most of the prediction approaches can be categorized as one of the naive, parametric, non-parametric and hybrid methodologies. Since the experimental studies presented in different works often used datasets from the different road network, it’s difficult to give a fair comparison.

Moreover, we find most of the existing prediction approaches are only suitable for small scale road network. It’s neither computationally bearable nor necessary to implement a large neural network for the national high way system. In this research, rather than treating the whole road network in one model, we divide the network into different disjoint regions using the k-means algorithm. The distance measure in the k-means algorithm is a convex combination of a time series distance and Euclidean distance. As a heuristic algorithm for clustering, k-means has some drawbacks, such as there no guarantee for optimal classification.

After grouping the edges of the high way network into different closely related regions, we employ several existing models to do the prediction for each region. Based on the most common evaluation criteria, we compare the performance of these models. In fact, we find for most of the areas, the simple LSTM neural network is
enough for prediction. Even compared with advanced DCRNN model, there is no much room for improvement.

5.2 Future work

There are several potential directions for future research. First, in this project, we divide the whole road network into different disjoint areas. Then the prediction models are applied for each region. However, the interactions among the boundaries of different regions are neglected. To avoid this drawback, fuzzy clustering methods can be employed to eliminate the clear boundaries between different regions. Another interesting idea is to incorporate the clustering into a neural network, therefore to reduce the parameters required.

Time series clustering is one of the most common methods for finding hidden patterns or similarity in multivariate time series data. By clustering, we may avoid much unnecessary computation. like the time series clustering method, Principal Component Analysis (PCA) is another powerful dimensionality reduction technique. Using PCA, we can transfer the whole network time series into several uncorrelated parts. The prediction models can then be utilized for the principal component of the network, rather than for every sensor. It’s believable that the prediction model associated with PCA will be a promising direction.
Bibliography


