EPFL

Thèse n° 8722

Multivariate time series forecasting for freeway networks

Présentée le 12 octobre 2022

Faculté de l'environnement naturel, architectural et construit
Laboratoire de systèmes de transports urbains
Programme doctoral en génie électrique

pour l'obtention du grade de Docteur ès Sciences

par

Semin KWAK

Acceptée sur proposition du jury

Prof. A. M. Alahi, président du jury
Prof. N. Geroliminis, directeur de thèse
Prof. F. Pereira, rapporteur
Prof. L. Sun, rapporteur
Prof. O. Fink, rapporteuse

École polytechnique fédérale de Lausanne

2022

Acknowledgements

I sincerely thank Nikolas for welcoming me from the first day we met with a big smile and didn't lose the spirit to the end. It was a great pleasure to work with him, who has excellent intuitions and gives maximal freedom in research at the same time. Also, profound thanks to Pascal, who always encourages and gives compliments even for small achievements I had. It was significant momentum for me to dig into a topic more deeply. It would not be possible to write Chapter 3 without his efforts.

LUTers are also a big part of my doctoral life. I want to give thank all my lab members. Their feedback and the time we shared were priceless for me. Also, special thanks to Christine, who gave great support for all administration works and showed me how to enjoy life properly and elegantly.

Great appreciation should go to my parents. They gave me infinite love to me. Sincerely appreciate their unconditional support and trust.

Without a doubt, the enormous thanks should go to Jenna. She motivated me to study in this beautiful country, enabling me to meet/work with all the fantastic people. It would not be possible to finish my doctoral study without her.

Also, I sincerely thank all my friends who support me here in and outside Switzerland.

Lausanne, 27 June 2022 Semin Kwak

Abstract

This dissertation introduces traffic forecasting methods for different network configurations and data availability. Chapter 2 focuses on single freeway cases. Although its topology is simple, the non-linearity of traffic features makes this prediction still a challenging task. We propose the dynamic linear model (DLM) to approximate the non-linear traffic features. Unlike a static linear regression model, the DLM assumes that its parameters change over time. We design the DLM with time-dependent model parameters to describe the spatiotemporal characteristics of time-series traffic data. Based on our DLM and its model parameters analytically trained using historical data, we suggest the optimal linear predictor in the minimum mean square error (MMSE) sense. We compare our prediction accuracy by estimating expected travel time based on the traffic prediction for freeways in California (I210-E and I5-S) under highly congested traffic conditions with other methods: the instantaneous travel time, k-nearest neighbor, support vector regression, and artificial neural network. We show significant improvements in accuracy, especially for short-term prediction.

Chapter 3 aims to generalize the DLM to extensive freeway networks with more complex topologies. Most resources would be consumed to estimate unnecessary spatiotemporal correlations if the DLM was directly used for a large-scale network. Defining features on graphs relaxes such issues by cutting unnecessary connections in advance based on predefined topology information. Exploiting the graph signal processing, we represent traffic dynamics over freeway networks using multiple graph heat diffusion kernels and integrate the kernels into DLM with Bayes' rule. We optimize the model parameters using Bayesian inference to minimize the prediction errors and, consequently, determine the mixing ratio of the two models (heat diffusion kernels and DLM). Such mixing ratio strongly depends on training data size and data anomalies, which typically correspond to the peak hours for traffic data. The proposed model demonstrates prediction accuracy comparable to state-of-the-art deep neural networks with lower computational effort. It notably achieves excellent performance for long-term prediction through the inheritance of periodicity modeling in DLM.

Chapter 4 proposes a deep neural network model to predict traffic features on large-scale freeway networks. These days, deep learning methods have heavily tackled traffic forecasting problems of freeway networks because they are outstanding to learn highly complex correlations between variables both in time and space, which the linear models might be limited

to. Adopting a graph convolutional network (GCN) becomes a standard to extract spatial correlations; therefore, most works have achieved great prediction accuracy by implanting it into their architecture. However, the conventional GCN has the drawback that receptive field size should be small, i.e., barely refers to traffic features of remote sensors, resulting in inaccurate long-term prediction. We suggest a forecasting model called two-level resolutions deep neural network (TwoResNet) that overcomes the limitation. It consists of two resolution blocks: The low-resolution block predicts traffic on a macroscopic scale, such as regional traffic changes. On the other hand, the high-resolution block predicts traffic on a microscopic scale by using GCN to extract spatial correlations, referring to the regional changes produced by the low-resolution block. This process allows the GCN to refer to the traffic features from remote sensors. As a result, TwoResNet achieves competitive prediction accuracy compared to state-of-the-art methods, especially showing excellent performance for long-term predictions.

Key words: Traffic forecasting, multivariate time series forecasting, dynamic linear model, graph heat diffusion, Bayesian inference, two-level resolution network.

Résumé

Cette thèse présente des méthodes de prévision du trafic routier pour différentes configurations de réseau à l'aide de données de disponibilité variable. Le chapitre 2 se concentre particulièrement sur les cas avec une seule autoroute. En dépit de la simplicité de cette topologie examinée, la difficulté de la prédiction réside dans les caractéristiques non linéaires du trafic. Nous proposons ainsi le modèle linéaire dynamique (DLM) pour approximer ces caractéristiques. Contrairement à un modèle de régression linéaire statique, le DLM considère que les paramètres varient dans le temps. Ainsi, dans le but de décrire les caractéristiques spatio-temporelles des données de trafic des séries chronologiques, nous considérons les paramètres du DLM comme dépendant du temps. En se basant sur le DLM et sur ses paramètres résultant de l'entraînement analytique du modèle à l'aide des données historiques, nous suggérons le prédicteur linéaire optimal qui minimise l'erreur quadratique moyenne (MMSE). Nous utilisons cette méthode pour estimer le temps de trajet prévu sur la base des prévisions de trafic pour les autoroutes de Californie (I210-E et I5-S) se trouvant dans des conditions très encombrées, et nous comparons les résultats de nos prévisions avec ceux obtenus par d'autres méthodes : le temps de trajet instantané, le k-plus proche voisin, la régression du vecteur de support, et le réseau de neurones artificiels. Nous montrons des améliorations significatives dans la précision, notamment pour les prévisions à court terme.

Le chapitre 3 vise à généraliser le DLM à des réseaux autoroutiers étendus avec des topologies plus complexes. Si nous utilisons le DLM directement dans un réseau à grande échelle, nous risquons de consommer la plupart des ressources pour estimer des corrélations spatio-temporelles inutiles. En définissant les caractéristiques des graphiques représentant nos réseaux, nous atténuons ces problèmes en supprimant les connexions inutiles à l'avance sur la base de données topologiques prédéfinies. À travers l'exploitation graphique du signal, nous représentons la dynamique du trafic sur les réseaux autoroutiers à l'aide de plusieurs noyaux de diffusion de chaleur de graphes, et nous intégrons ensuite ces noyaux dans le DLM en se fondant sur la règle de Bayes. Nous optimisons les paramètres du modèle en utilisant l'inférence bayésienne pour minimiser les erreurs de prédiction et nous déterminons par conséquent le rapport de mélange des deux modèles (noyaux de diffusion de chaleur et DLM). Un tel rapport de mélange dépend largement de la taille des données d'entraînement et des anomalies de données. Celles-ci correspondent généralement aux heures de pointe dans les données de trafic. Le modèle proposé démontre une précision prédictive comparable

aux réseaux de neurones profonds avancés, mais avec un effort de calcul inférieur. Il réalise notamment d'excellentes performances de prédiction à long terme grâce à l'héritage de la modélisation de périodicité du DLM.
Le chapitre 4 propose un modèle de réseau neuronal profond permettant de prévoir les caractéristiques du trafic sur les réseaux autoroutiers à grande échelle. De nos jours, les modèles d'apprentissage en profondeur se sont fortement attaqués aux problèmes de prévision du trafic des réseaux autoroutiers vu leur capacité remarquable pour apprendre des corrélations très complexes entre des variables à la fois dans le temps et dans l'espace. Quant aux modèles linéaires, ils pourraient être limités à cet égard. L'adoption d'un réseau convolutif pour graphe (GCN) devient ainsi une norme pour l'extraction de corrélations spatiales. De ce fait, la plupart des œuvres ont obtenu une grande précision prédictive en l'implantant dans leur architecture. Toutefois, l'inconvénient du GCN conventionnel est que la taille du champ récepteur doit être petite, ce qui signifie qu'il renvoie à peine aux caractéristiques de trafic des capteurs à distance. Ceci entraîne une prédiction à long terme inexacte. Nous proposons un modèle de prévision appelé réseau de neurones profonds à résolution à deux niveaux (TwoResNet) qui surmonte cette limitation. Il se compose de deux blocs de résolution : D'une part, le bloc à basse résolution prévoit le trafic à une échelle macroscopique, tels que les changements dans le trafic régional. D'autre part, le bloc à haute résolution prévoit le trafic à l'échelle microscopique à l'aide de GCN pour extraire les corrélations spatiales en se référant aux changements régionaux produits par le bloc à basse résolution. Ce processus permet au GCN de se reporter aux caractéristiques du trafic provenant de capteurs à distance. En conséquence, TwoResNet atteint une précision de prédiction concurrentielle par rapport aux méthodes avancées, présentant en particulier d'excellentes performances pour les prévisions à long terme.

Mots clefs : prévision de trafic, prévision de séries temporelles multivariées, modèle linéaire dynamique, diffusion de chaleur par graphe, inférence bayésienne, réseau de résolution à deux niveaux.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background and motivation

Traffic forecasting is one of the essential elements in intelligent transportation systems (ITS). Accurate prediction enables traffic operators to control road traffic precisely, ultimately providing optimal traffic flow in road networks. It also benefits individuals to estimate the expected travel time for their trips.

Adequate infrastructure and data acquisition frameworks must be in place beforehand for successful traffic forecasting. Installing cameras or sensors on road networks is a common approach to collecting traffic data. Although it requires a significant investment in building sensor networks, it allows consistent data acquisition once established. On the other hand, with low-cost and widespread electronic devices, large-scale data can also be obtained from drivers' GPS information. However, in case of the GPS information, a significant noise in the data can complicate its pre-processing. Furthermore, many difficulties exist in regularizing sporadic data using map-matching algorithms [1] as well as challenges in data acquisition due to high uncertainty.

Once the data-collection infrastructure is established, extracting maximal information from data should follow. Extracting temporal correlations from traffic data is one of the essentials for successful forecasting. In general, traffic features collected from a sensor has a smooth pattern and strong periodicity. In other words, the traffic state after some time is closely related to the current traffic state (smoothness). We also expect similar patterns on daily and weekly bases (periodicity).

How to combine these two features appropriately is one of the key challenges in designing a traffic predictor. Two naive predictors can be easily introduced, considering either the smoothness or the periodicity: An instantaneous predictor forecasts future traffic based on the traffic at the current time. On the other hand, a k-nearest neighbor predictor picks k-similar patterns from the current traffic and predicts by averaging the patterns. These two simple predictors are often used as baselines since they achieve decent accuracy for short

and long-term predictions, respectively. As representative models that properly combine these two assumptions, autoregressive (AR)-based models [2], which are mathematically rigorously defined, predict traffic states by giving more weight to smoothness than periodicity. On the other hand, heuristic models are more optimized for predicting traffic volume and have excellent performance, but the model's flexibility is limited due to the ad-hoc design.

The spatial correlation of traffic is another puzzle to solve for successful prediction. For instance, a shock wave theory explains that congestion spreads and disappears through a network, rather than staying at the origin and disappearing. Therefore, when a sensor detects a traffic change, another nearby sensor detects the shock wave of the traffic change with a time lag. This means that the values of surrounding sensors can be an essential clue when predicting the future value.

Extracting spatial correlation itself is challenging when the number of correlations is large. The difficulty is proportional to the size of the freeway network since the number of correlations exponentially increases with the number of sensors. In the case of a single freeway, as a result, satisfactory prediction performance can be obtained even by simply using an AR model with vectorized input without unique spatial modeling [3]. On the other hand, spatial correlations should be extracted more carefully for complex networks; otherwise, unnecessary correlations consume most resources. Most studies often start by assuming that the topology of a freeway network is significantly related to the spatial correlation of its sensors. For example, traffic flows at two different points in the city center closely influence each other. But at the same time, flow or density in a very distant suburb may not be correlated much. Based on the assumption, the topology of a network is exploited as a priori information independent of actual traffic characteristics.

A graph is a mathematical expression for defining a selective spatial correlation. Defining a freeway network as a graph can improve the prediction accuracy using rigorous theories such as a graph theory and graph signal processing [4]. For example, the diffusion model of graph signals generalizes a link-level shock wave theory to the whole network level. More precisely, it describes the solution of a differential equation that expresses the temporal change of a graph signal as a spatial difference from neighboring signals. As a result, this diffusion model can precisely approximate traffic volume changes over time in freeway networks.

Although the changes in traffic volume have been successfully approximated through diffusion models, we also need to integrate these data-independent models with data-driven models as a next challenge. For example, they can be fused by using Bayes' theorem with some parameters. After that, these parameters are estimated probabilistically using historical data through a methodology called Bayesian inference [5]. The Bayesian inference has a unique feature: it infers all the parameters optimally, while many machine learning models estimate them through time-consuming algorithms.

Recently, many studies have implemented neural networks to extract spatiotemporal correlations. Although a neural network is much more complex than all the models above, an easy

access to data and the rapid development of deep learning techniques brings a surge in their usage. Neural networks have made great strides for the following reasons: (1) Any arbitrary functions can be theoretically expressed even with elementary modeling. (2) Synthesizing non-linear functions with a neural network architecture makes it possible to numerically estimate model parameters. (3) Although this numerical process is demanding, it can be overcome with proper approximation and dramatically increased computing power. (4) Most mathematical steps are automatized by great standard libraries.

Most neural network-based traffic forecasters consist of separate blocks that extract temporal and spatial correlations and concatenate them to extract complex spatiotemporal correlations. Temporal correlations are mainly extracted using a recurrent neural network [6], a temporal convolutional network [7], or an attention network [8]. In contrast, spatial correlations are extracted by modeling them with message-passing graph neural network (GNN) variations [9], such as graph convolutional and graph attention networks [10]. Although neural network-based forecasters have achieved state-of-the-art prediction accuracy, there is still a room for improvements since the GNN is incomplete to extract latent spatial correlations. A well-known challenge to extracting spatial correlations with conventional GNN is the limited range of correlations each node can refer to [11], resulting in a degradation of long-term prediction accuracy: Since congestion takes time to propagate from one point to another, referring to traffic features from a remote sensor where congestion is detected is necessary to capture such congestion.

1.2 Objectives

The overarching objective of this thesis is to build optimal traffic forecasters for freeway networks, tackling the challenges mentioned above. The objectives fall into two main categories: (1) Designing a predictor for simple freeway networks (more precisely, a single freeway). (2) Extending the design for complex freeway networks. The detailed objectives of each category according to dissertation structure are listed as follows:

1. Designing a predictor for simple freeway networks
    (a) To design a model that explicitly expresses smoothness and periodicity of traffic.
    (b) To make the model simple enough to derive all the estimation process in a closed form.
    (c) To provide accurate prediction results compared to the two baselines.
2. Extending the design for complex freeway networks
    (a) To generalize the simple model, making it works for extensive freeway networks.
    (b) To develop a neural network model that learns more complex spatiotemporal correlations, expecting excellent long-term prediction performance.

1.3 Contributions

The main contributions of this dissertation are summarized hereafter.

Designing a predictor for simple freeway networks: The main contribution in this part is to propose a piecewise linear model called the dynamic linear model (DLM) that effectively represents nonlinear traffic dynamics, explicitly implying the strong periodicity of traffic in the model. This piecewise linear model shows excellent predictive performance, resulting in accurate estimation for travel time. By exploiting the simplicity of the linear model, we estimate the model's parameters in a closed form with historical data giving more importance to the latest data. In addition, we also derive a method to update the analytic solution with new data in a mathematically rigorous way.

Extending the design for complex freeway networks: One of the main contributions in this part is to generalize the DLM into more extensive freeway networks. We propose a novel method that successfully integrates topological information into the DLM. We model the propagation of congestion by decomposing it with different propagation speeds: We suggest a parameterized mixture of diffusion kernels whose each kernel expresses a different propagation speed of the congestion on the predefined topology. After that, we fuse the topology-based congestion modeling with DLM, which is a data-driven congestion modeling by Bayes' theorem. We also derive an inference method for hyperparameters, exploiting Bayesian inference. The training time required for inference is minimal since we successfully derive the majority of inference steps analytically. As a result, the proposed model shows a great prediction accuracy on extensive freeway networks together with sufficient interpretability. In other words, the trained model is straightforward to analyze, unlike other deep neural network-based models. We propose a neural network architecture that captures spatiotemporal correlations at different scales as another main contribution: The proposed model captures spatiotemporal correlations at a macroscopic level with the low-resolution block, while the high-resolution block captures them at a microscopic level. We introduce a clustering method to divide complex freeway networks into subgraphs to get regional trends, considering proximity and correlations simultaneously. As a result, the proposed model achieves a competitive prediction accuracy compared to reliable state-of-the-art works. For the long-term predictions, we obtained remarkable improvements compared to a neural network that learns road connectivity directly from data.

1.4 Thesis structure

This dissertation consists of 5 chapters. The main chapters are organized into two parts. The first part is chapter 2, which focuses on developing a traffic forecasting model for simple freeway networks. The second part includes chapters 3 and 4, suggesting forecasting models for extensive freeway networks. Each chapter starts with detailed literature reviews corresponding

to the contents of the chapter.

Chapter 2 introduces the dynamic linear model (DLM) for simple freeway networks to model nonlinear traffic dynamics. An estimation method of the model parameter is suggested. After that, a method is introduced to predict future traffic and the expected travel time based on the DLM with estimated parameters. The results and analysis of the prediction accuracy are followed. Chapter 2 is a stand-alone article published as:

- Kwak, Semin, and Nikolas Geroliminis. “Travel time prediction for congested freeways with a dynamic linear model." IEEE Transactions on Intelligent Transportation Systems 22.12 (2020): 7667-7677.

Chapter 3 introduces a generalization method of the DLM into extensive freeway networks. Essential mathematical definitions are introduced at first, and a method is explained to model congestion propagation with graph diffusion kernels only with topological information. A method to integrate the topology-dependent congestion propagation model into the DLM is explained. It is described to infer all the necessary parameters by Bayesian inference and a method to predict future traffic. Detailed analyses of the prediction results follows. Chapter 3 is a stand-alone article published as:

- Kwak, Semin, Nikolas Geroliminis, and Pascal Frossard. “Traffic signal prediction on transportation networks using spatio-temporal correlations on graphs." IEEE Transactions on Signal and Information Processing over Networks 7 (2021): 648-659.

Chapter 4 proposes a neural network architecture that predicts future traffic on extensive freeway networks. The model architecture is introduced in a top-down approach. First, the low-resolution block is introduced, which predicts future traffic at a macroscopic level, followed by explanations of the interior components of the blocks. After that, the high-resolution block is explained that forecasts upcoming traffic at a microscopic level, then the necessary details of the block are explained. Test results on benchmark datasets and ablation study are presented. Chapter 4 is based on a preliminary result:

- Semin Kwak, Danya Li and Nikolas Geroliminis. "TwoResNet: Two-level resolution neural network for traffic forecasting of freeway networks". In: (Macau, China, Otc. 8-12, 2022). IEEE ITSC 2022 - 25th IEEE International Conference on Intelligent Transportation Systems, 2022 (submitted).

Chapter 5 explains the conclusion of the dissertation and future research.

2 Traffic forecasting with Dynamic linear model

This chapter is written with its own notations, and the content of this chapter is available as a published paper:

> Kwak, Semin, and Nikolas Geroliminis. "Travel time prediction for congested freeways with a dynamic linear model." IEEE Transactions on Intelligent Transportation Systems 22.12 (2020): 7667-7677.

2.1 Introduction

Travel time prediction is one of the essential features to support successful Intelligent Transportation Systems (ITS). An accurate prediction of travel time not only helps travelers to make decisions about their trips but also enables traffic operators to develop successful control strategies. This necessity has engaged many researchers on the topic of travel time forecasting despite the vast amount of already existing literature.

The methods for predicting travel time can be categorized into model-based and data-driven approaches [12]. The model-based methods predict future traffic parameters (e.g., occupancy, flow, or speed) by building a traffic model, such as the Cell Transmission Model [13], [14], the queuing theory [13], [15], [16], or macroscopic traffic flow model [17]. These model-based methods provide a straightforward interpretation of the predicted results because of their physical intuition, such as flow dynamics.

The data-driven methods, on the other hand, predict travel time by extracting specific features from traffic data. Common data-driven methods include Linear Regression [18], [19], Autoregressive models [20]–[23], Kalman Filtering [24]–[27] and Bayesian inference [28], [29]. These methods predict travel time by assuming that all the data satisfies a certain probabilistic distribution.

Furthermore, the increased accessibility to traffic data and the improved computing power in these days allow researchers to develop more sophisticated data-driven algorithms, such as Support vector regression (SVR) [30]–[32], Artificial neural networks (ANN) [33]–[40], Long Short-Term Memory Network [41], [42] and Ensemble learning [43]–[45].

Conversely, the travel time predictors can also be categorized into *direct* and *indirect* methods. Direct methods contain a straightforward approach to minimize the error in predicted travel time [18]–[23], [25]–[36], [40], [41], [44]. The main advantage of the direct methods lies in their simplicity since they take into account only the travel time as an output. However, the prediction performance can also be degraded as all the complex traffic characteristics are assumed to be reflected in travel time. Another limitation of these methods is that they require separate models for different circumstances; for example, when the departure time or the origin location change a new model has to be trained for that exact setting.

In contrast, the indirect methods estimate travel time by predicting future traffics first, such as velocity or occupancy field. Then they use the predicted traffic states to estimate travel time [24], [43], [46], [47]. By definition, the predicted traffic states can also be re-used to predict those for the next horizon. Contrarily to the direct methods, predicting future traffic parameters allows the model to estimate a travel time for any scheduled departure time or space, which makes the indirect method more versatile.

In this chapter, we suggest a method based on a dynamic linear model to predict the velocity field and therefore travel time, which falls into the intersection of the indirect method and the data-driven approach. Using historical data, we analytically find the model parameters in the least-squares sense. We compare the proposed method with four other predictors that are used in the literature: the instantaneous travel time, the k-nearest neighbor [46], artificial neural networks [33], and the support vector regression [30]. Our comparison shows that the proposed method has a great potential to improve the short-term prediction accuracy as well as to become a versatile tool in various traffic situations with this stand-alone model.

2.2 Method

2.2.1 Dynamic linear model

We suggest a dynamic linear model for speed and travel time prediction. The dynamic characteristics allow the model to extract temporal features of the parameters of interest (velocity fields in our case). The model describes a linear relationship between velocities at a specific time t_k and the next step time t_{k+1} using the following equation:

$$\mathbf{v}_{k+1}^d = H_k \mathbf{v}_k^d + \mathbf{n}_k^d, \ \forall d, \ \forall k \in \{0, \dots, K-1\}. \tag{2.1}$$

Here the vector $\mathbf{v}_k^d$ refers to a velocity vector at time t_k on day d, which can be expressed as follows:

$$\mathbf{v}_k^d = \begin{bmatrix} v^d(x_1, t_k) \\ \vdots \\ v^d(x_M, t_k) \end{bmatrix} \in \mathbb{R}^{M\times 1}, \tag{2.2}$$

where the constant M represents the number of measured velocities on different locations on a freeway of interest, for example with data from loop detectors. We define a velocity field as a scalar function of time t and position x:

$$v^d(x,t) \in \mathbb{R}, \tag{2.3}$$

where each point of the velocity field represents a measured velocity value.

In Eq. (2.1), the second vector $\mathbf{n}_k^d$ on the right-hand side refers to a noise vector which we assume to follow a Gaussian distribution with a zero mean and a variance σ^2 under independent and identically distributed (i.i.d.) conditions, i.e.:

$$\mathbf{n}_k^d \sim \mathcal{N}\left(\mathbf{0}_M, \sigma^2 I_M\right),\ \forall k \in \{0,\dots,K-1\}, \tag{2.4}$$

where $\mathbf{0}_M$ and I_M are the vectors with all zero entities and the identity matrix of size M, respectively.

Matrix H_k in Eq. (2.1) is a transition matrix, which represents a linear relationship between the two velocity vectors $\mathbf{v}_k^d$ and $\mathbf{v}_{k+1}^d$ according to the time t_k and t_{k+1}. In particular, the diagonal elements of H_k describe a direct temporal relationship at each specific location, whereas the off-diagonal terms of H_k contain the spatio-temporal relationship between two consecutive velocity fields. The first aim of this chapter is to find an analytical solution of the transition matrix H_k for every possible time t_k so that we build a dynamic transition matrix.

The model presented in Eq. (2.1) suggests three important factors. First, the velocity vector $\mathbf{v}_k^d$ is *linearly* transformed to the vector $\mathbf{v}_{k+1}^d$ with an additive Gaussian noise. The linearity and the Gaussian noise assumption allow the transition matrices H_k for every k to be trained analytically, which will be discussed in the next section. Secondly, the transition matrix is defined at each time unit such that the model captures a non-linear traffic flow over time even though it is based on a linear regression model for a given period. Lastly, the transformation matrix of two consecutive time steps k and $k+1$ is set regardless of different traffic profiles, which means that the matrix H_k does not depend on, for example, the days of a week. We will show later that this framework captures well traffic conditions of different days.

2.2.2 Estimation of model parameters

We shall estimate the transition matrices H_k for all k values with historical data set using the least-squares method. Within a set of days $\mathbb{D}$ we choose for estimation (or we call it a day set), Eq. (2.1) can be extended as

$$V_{k+1}^{\mathbb{D}} = H_k V_k^{\mathbb{D}} + N_k^{\mathbb{D}}, \tag{2.5}$$

where the matrix $V_k^{\mathbb{D}}$ is a time-velocity matrix defined for the day set $\mathbb{D}$ for a specific time t_k as a collection of all velocity vectors corresponding to the same time index within $\mathbb{D}$, i.e.:

$$V_k^{\mathbb{D}} = \begin{bmatrix} \mathbf{v}_k^{d_1} & \mathbf{v}_k^{d_2} & \dots & \mathbf{v}_k^{d_{|\mathbb{D}|}} \end{bmatrix} \in \mathbb{R}^{M \times |\mathbb{D}|}, \ \forall d_i \in \mathbb{D}. \tag{2.6}$$

Here, the operator $|\cdot|$ of a set represents the cardinality (the number of elements) of the set. Therefore, the number of the rows and columns represents the data dimension and the size of a day set, respectively.

From Eq. (2.5), we shall estimate the transition matrix using the least-squares method, which is also equivalent to the solution of the maximum likelihood method since we assume i.i.d. Gaussian noise [48]. Therefore, the optimization problem can be stated as:

$$\underset{H_k}{\text{minimize}} \left\| V_{k+1}^{\mathbb{D}} - H_k V_k^{\mathbb{D}} \right\|_F^2, \tag{2.7}$$

where the operator $\|A\|_F = \sqrt{\text{tr}\left(AA^\top\right)}$ and $\text{tr}\left(AA^\top\right)$ indicates a sum of the all diagonal elements of a matrix $AA^\top$.

In order to prevent an ill-posed problem and to give priority to more recent data for better prediction, we introduce an adaptive matrix regularization term with a regularization parameter ρ and a forgetting factor λ, which is recursively multiplied to old data set, to Eq. (2.7) as follows:

$$\underset{H_k}{\text{minimize}} \ \rho\lambda^{|\mathbb{D}|} \|H_k\|_F^2 + \left\| \left(V_{k+1}^{\mathbb{D}} - H_k V_k^{\mathbb{D}}\right) \Lambda_{|\mathbb{D}|}^{\frac{1}{2}} \right\|_F^2, \tag{2.8}$$

where the diagonal matrix Λ_N is defined as follows with the forgetting factor λ:

$$\Lambda_N = \begin{bmatrix} \lambda^{N-1} & 0 & \cdots & 0 \\ 0 & \lambda^{N-2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix}. \tag{2.9}$$

The first term in Eq. (2.8) is the regularization term; its major role is to prevent the transition matrices from overfitting to a small training data set. The term also allows reliable estimation of the transition matrix numerically, which is described in Appendix A.1.2.

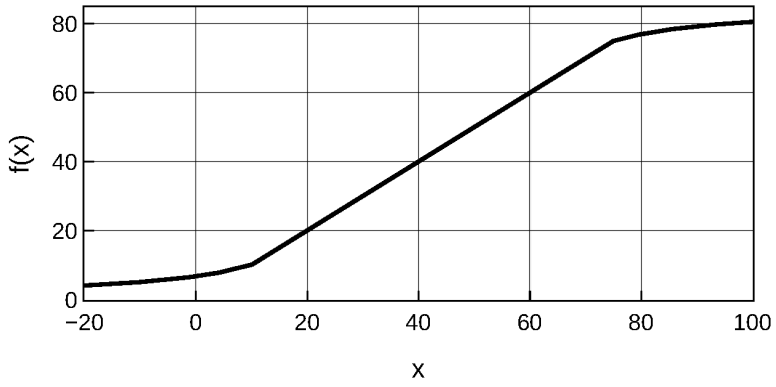


Figure 2.1: Post-processing function for the freeways studied in this chapter. The function makes the output values fall within a reasonable speed range, i.e., $0 \leq f(x) \leq 85$.

The forgetting factor, on the other hand, decreases the weight of old data exponentially during the recursive training process. For instance, when $\lambda = 0.995$, a set of data a year ago is penalized by the factor of $(0.995)^{365} = 0.16$. This forgetting factor also allows the regularization term to vanish, adapting to the size of the training set since the term converges to zero when the number of elements in $\mathbb{D}$ is getting bigger. The modified problem in Eq. (2.8) will be equivalent to the original problem of Eq. (2.7) when we set $\rho = 0$ and $\lambda = 1$, which means no regularization and no forgetting process.

The optimization problem in Eq. (2.8) can be analytically solved, and we derive it in Appendix A.1.1. Here we present the solution:

$$\bar{H}_k^{\mathbb{D}} = V_{k+1}^{\mathbb{D}} \Lambda_{|\mathbb{D}|} \left(V_k^{\mathbb{D}}\right)^\top \left(V_t^{\mathbb{D}} \Lambda_{|\mathbb{D}|} \left(V_k^{\mathbb{D}}\right)^\top + \rho \lambda^{|\mathbb{D}|} I_M\right)^{-1}, \tag{2.10}$$

where the notation $\bar{A}$ represents an estimator of A.

One popular property of the solutions of the least squares problem is that they can be updated with new observations [48]. The updating method not only prevents increasing memory size, but also makes computation time consistent since the procedure only needs a pre-trained model and a new observation for an update. This property can be essential to support accurate travel time prediction because the system should be up-to-date with time. We describe the implementation of an updating algorithm for Eq. (2.10) in Appendix A.1.3.

2.2.3 Velocity prediction

With the transition matrices $\bar{H}_k^{\mathbb{D}}$ introduced in the previous section, we can now predict the velocity vectors for traffic forecasting, which is the second aim of this chapter. We start by setting a notation of a predictor for i-step ahead at time step k:

$$\mathbf{v}_{k+i|k}^{\tilde{d}} \text{ for } i = 1, 2, \cdots \text{ and } \tilde{d} \notin \mathbb{D}. \tag{2.11}$$

Assuming that the trained transition matrix is close enough to the truth i.e., $\bar{H}_k^{\mathbb{D}} \approx H_k$ for all k, the velocity vector $\mathbf{v}_{k+i}^{\tilde{d}}$ is written as follows using Eq. (2.1):

$$\mathbf{v}_{k+i}^{\tilde{d}} = \bar{H}_{k+i-1}^{\mathbb{D}} \mathbf{v}_{k+i-1}^{\tilde{d}} + \mathbf{n}_{k+i-1}^{\tilde{d}} \tag{2.12}$$

$$= \bar{H}_{k+i-1}^{\mathbb{D}} \left(\bar{H}_{k+i-2}^{\mathbb{D}} \mathbf{v}_{k+i-2}^{\tilde{d}} + \mathbf{n}_{k+i-2}^{\tilde{d}} \right) + \mathbf{n}_{k+i-1}^{\tilde{d}} \tag{2.13}$$

$$\vdots$$

$$= \bar{H}_{k+i-1\leftarrow k}^{\mathbb{D}} \mathbf{v}_k^{\tilde{d}} + \mathbf{n}_{k+i-1\leftarrow k}^{\tilde{d}}, \tag{2.14}$$

where

$$\bar{H}_{k+i-1\leftarrow k}^{\mathbb{D}} = \prod_{j=1}^{i} \bar{H}_{k+i-j}^{\mathbb{D}}, \tag{2.15}$$

$$\mathbf{n}_{k+i-1\leftarrow k}^{\tilde{d}} = \sum_{j=1}^{i-1} \bar{H}_{k+i-j}^{\mathbb{D}} \mathbf{n}_{k+i-j-1}^{\tilde{d}} + \mathbf{n}_{k+i-1}^{\tilde{d}}. \tag{2.16}$$

The noise vector in Eq. (2.14) follows a zero mean Gaussain vector with a covariance Σ since a linear combination of zero mean Gaussian random variable follows another zero mean Gaussian random variable [49]. As a result,

$$\mathbf{v}_{k+i}^{\tilde{d}} \sim \mathcal{N}\left(\bar{H}_{k+i-1\leftarrow k}^{\mathbb{D}} \mathbf{v}_k^{\tilde{d}}, \Sigma \right). \tag{2.17}$$

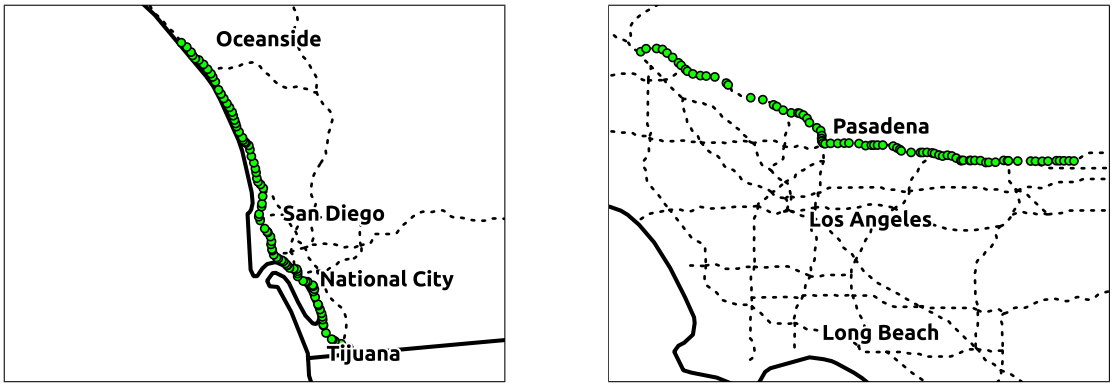
We choose a predictor as the maximizer of the above density function:

$$\mathbf{v}_{k+i|k}^{\tilde{d}} = \bar{H}_{k+i-1\leftarrow k}^{\mathbb{D}} \mathbf{v}_k^{\tilde{d}}. \tag{2.18}$$

This predictor is also an optimal estimator of the linear minimum mean square error (LMMSE) (Appendix A.1.4). Therefore, Eq. (2.18) shows that the best linear predictor $\mathbf{v}_{k+i|k}^{\tilde{d}}$ is the propagation of the current measurement $\mathbf{v}_k^{\tilde{d}}$ through the trained transition matrices from $\bar{H}_k^{\mathbb{D}}$ to $\bar{H}_{k+i-1}^{\mathbb{D}}$.

However, in rare cases, an unbounded solution $\mathbf{v}_{k+i|k}^{\tilde{d}}$ can have a negative speed or an unrealistically high speed due to the Gaussian noise assumption. This kind of wrong estimations severely distort the calculation of travel time. In order to correct this effect, we design a post-processing function $f(x)$:

$$f(x) = \begin{cases} b \cdot \frac{a(x-\tau_l)}{1+|a(x-\tau_l)|} + \tau_l & x < \tau_l \\ x & \tau_l \le x \le \tau_u \\ b \cdot \frac{a(x-\tau_u)}{1+|a(x-\tau_u)|} + \tau_u & x > \tau_u \end{cases}, \tag{2.19}$$



(a) Freeway I5-S (b) Freeway I210-E

Figure 2.2: Detector locations on the freeways I5-S and I210-E. The 88 loop detectors along the freeway I5-S and the 83 loop detectors along the freeway I210-E are used in this chapter (green dots on both figures). In the names of the freeways, S (south) and E (east) represent the direction of the freeways.

where the constants a and b are smoothing parameters, and τ_l and τ_u are threshold parameters. We empirically set the smoothing parameters a and b to be 0.05 and 10, respectively. We also have empirically chosen the lower threshold value τ_l and the upper threshold value τ_u as 10 and 75 (mph), respectively.

Figure 2.1 shows the post-processing function with the chosen parameter sets. The input x in this example is a velocity value. When x is below the lower threshold of 10, the function deflects the values to be always positive. When x is above the upper threshold of 75, the function makes the output converging to the upper limit, which is 85 miles per hour in our case. Between the two boundaries, it does not change the input value. We have tested different sets of the smoothing and threshold parameters and found that it has little impact on prediction results.

We apply this post-processing function in Eq. (2.19) to Eq. (2.18) recursively at each step of multiplication so that we can exclude the invalid estimations. The following shows the detailed procedure.

$$\begin{aligned}
\mathbf{v}_{k+1|k} &= f\left(\tilde{H}_k \mathbf{v}_k\right) \\
\mathbf{v}_{k+2|k} &= f\left(\tilde{H}_{k+1} \mathbf{v}_{k+1|k}\right) \\
&\vdots \\
\mathbf{v}_{k+i|k} &= f\left(\tilde{H}_{k+i-1} \mathbf{v}_{k+i-1|k}\right)
\end{aligned} \tag{2.20}$$

2.2.4 Travel time estimation

For estimating the travel time of a moving vehicle, we assume that the vehicle experiences a velocity field, which is a function of time t and space x, and we know the exact continuous velocity field $v(t,x)$. Then we can calculate the increment of time Δt after traveling a distance Δx as

$$\Delta t = \frac{1}{v(t,x)} \Delta x, \tag{2.21}$$

since $v = dx/dt$. Consequently, a travel time at time t_0 given a velocity field $v(t,x)$ is computed recursively as follows:

Algorithm 1 Numerical calculation of travel time

Input: the velocity field $v(t,x)$; the departure time and location, t_0 and x_0; the location of the destination x_M; and the space increment Δx

Output: the travel time

1: **Initialization:** $t \leftarrow t_0$, $x \leftarrow x_0$
2: **while** $x < x_M$ **do**
3: $\quad t \leftarrow t + \frac{1}{v(t,x)} \Delta x$
4: $\quad x \leftarrow x + \Delta x$
5: **end whilereturn** $t - t_0$

In reality, we only know a discretized velocity field instead of a continuous one. For our study, we know velocities at each sensor (every 0.7 miles on average) every 5 minutes. We generate the continuous velocity field by interpolating the discretized velocity field with linear bivariate B-spline curve fitting.

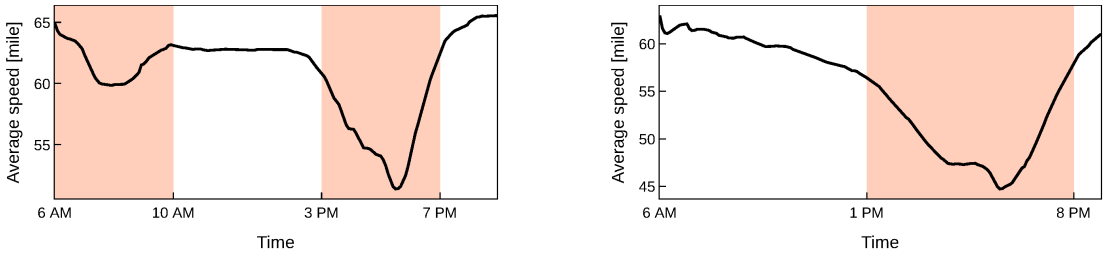
2.2.5 Performance measures for comparison with other methods

To measure the performance of our prediction, we use the absolute percentage error (APE) and the mean absolute percentage error (MAPE), which are defined as:

$$\mathrm{APE}(t) = 100 \cdot \left| \frac{a(t) - p(t)}{a(t)} \right|, \tag{2.22}$$

$$\mathrm{MAPE}(\mathbb{T}) = \frac{1}{|\mathbb{T}|} \sum_{t \in \mathbb{T}} \mathrm{APE}(t), \tag{2.23}$$

where the set $\mathbb{T}$ represents a set of time elements to examine. The values $a(t)$ and $p(t)$ are respectively the actual travel time and the predicted travel time when departed at time t. The MAPE estimates the mean deviation of estimation to the ground truth (i.e., the experienced travel time) in percentage (%) unit.



(a) Freeway I5-S (weekdays)

(b) Freeway I210-E (weekdays and Saturday)

Figure 2.3: Average speed by time of test set for each freeway. The peak periods are also defined as the area with color.

2.3 Results and Discussions

In this section, we employ the proposed method to predict traffic flow and thus travel time using real-world data. We examine the performance of the proposed method by comparing predicted travel time with that of other existing predictors.

2.3.1 Traffic data

We use traffic data of two different freeways in California having different traffic profiles: Freeway I5-S and Freeway I210-E. Along the two freeways, there are respectively 88 and 83 loop detectors within the area of our examination (Fig. 2.2). The total length of the corridor along I5-S is 58.33 miles, and that of I210-E is 52.14 miles. The loop detectors collect measurements (flow and occupancy data) every 30 second, and we use 5 minutes aggregated speed data, which is processed by the Caltrans Performance Measurement System (PeMS).

From PeMS, we extracted one-year traffic data of both freeways (2012 for I5-S and 2015 for I210-E) for experiments. We allocated the first 70% of traffic data (from January 1st to September 12th) as a training set, the next 15% of data as a validation set (from September 13th to November 11th), and the last 15% of data as a test set (from November 12th to December 31st) for all the experiments.

We have considered the traffic data from 6 AM to 9 PM only and divided the data into two groups: a peak period and an off-peak period (Fig. 2.3). Since the two freeways have very different traffic profiles, we have defined the peak and off-peak periods differently for each freeway. For Freeway I5-S, the peak period is defined as 6 - 10 AM (morning peak) and 3 - 7 PM (evening peak) on weekdays (from Mondays to Fridays); for Freeway I210-E, it is defined as 1 - 8 PM (afternoon peak) every day except Sundays. The off-peak periods are defined as a complementary set of the corresponding peak periods. Figure 2.3 illustrates them straightforwardly.

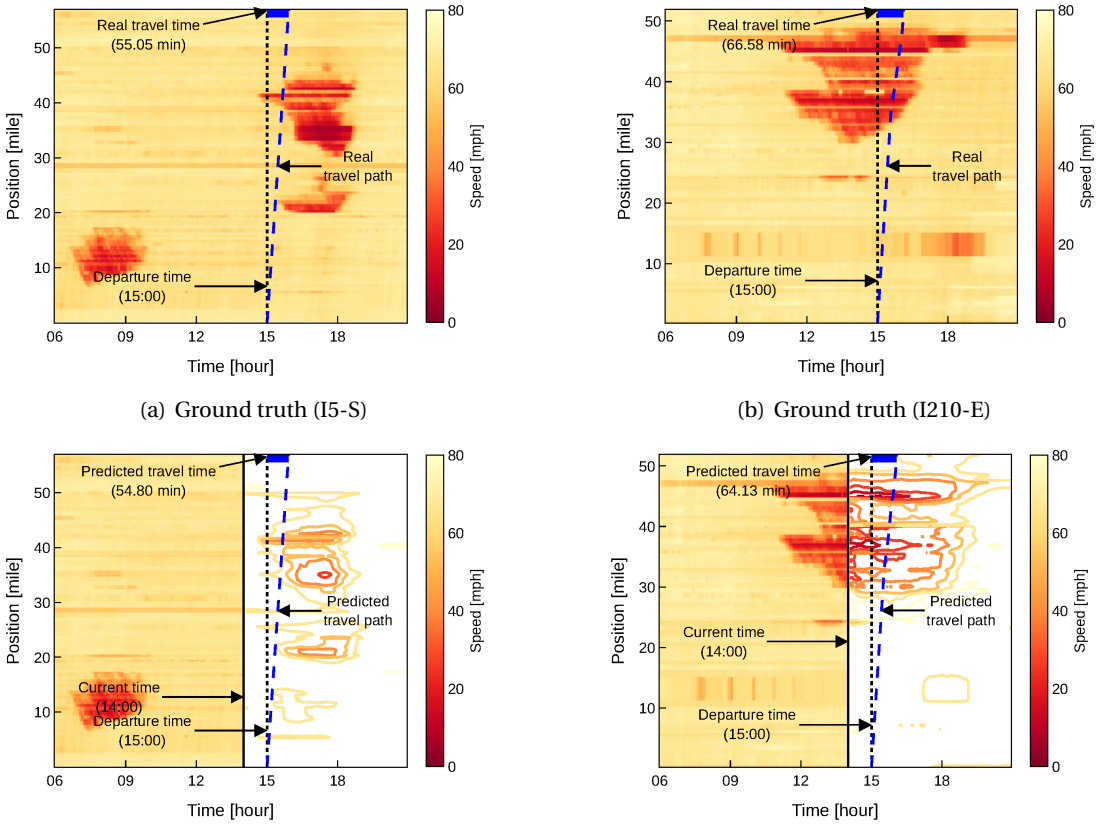
Table 2.1: Mean absolute percentage error (MAPE) on the validation sets for Freeways I5-S and I210-E with different hyper-parameter pairs

(a) Freeway I5-S

ρ \ λ	1.000	0.999	0.995	0.990	0.950
0	3.447	3.433	3.414	3.432	5.134
1	3.441	3.428	3.411	3.430	5.134
3	3.430	3.418	3.405	3.427	5.134
10	3.395	3.388	3.385	3.415	5.134
30	3.318	3.319	3.340	3.385	5.134
100	3.183	3.187	3.232	3.317	5.133
300	3.071	3.074	3.106	3.202	5.130
1000	2.996	2.987	2.982	3.045	5.119
3000	3.003	2.987	**2.936**	2.937	5.091
10000	3.244	3.192	3.043	2.926	5.003

(b) Freeway I210-E

ρ \ λ	1.000	0.999	0.995	0.990	0.950
0	4.842	4.879	5.048	5.280	7.456
1	4.858	4.896	5.063	5.290	7.455
3	4.848	4.888	5.058	5.288	7.454
10	4.816	4.859	5.037	5.275	7.453
30	4.747	4.793	4.986	5.239	7.453
100	4.641	4.673	4.865	5.143	7.451
300	**4.602**	4.622	4.729	4.998	7.449
1000	4.643	4.637	4.672	4.808	7.443
3000	4.806	4.781	4.718	4.745	7.427
10000	5.318	5.232	4.965	4.811	7.369



(a) Ground truth (I5-S)

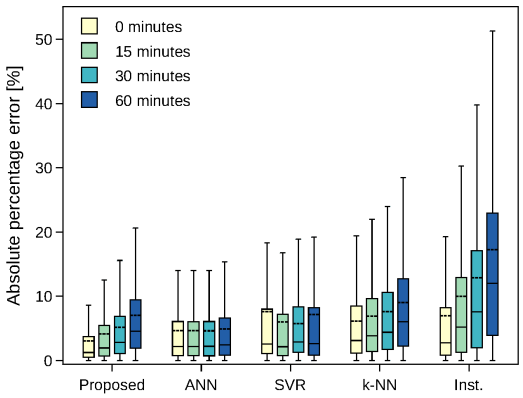
(b) Ground truth (I210-E)

(c) Prediction (I5-S)

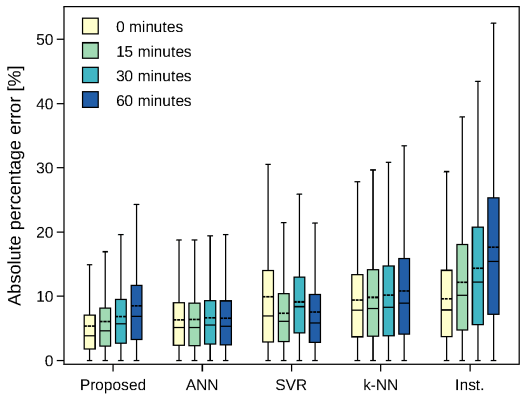
(d) Prediction (I210-E)

Figure 2.4: Velocity field of freeway I5-S on December 4th (Tuesday), 2012 and I210-E on December 25th (Friday), 2015. (a) and (b): The ground truths and (c) and (d): predicted velocity fields at 2 PM and afterwards using the proposed method, which is represented as the contour plots. The blue dashed line represents the travel path of a vehicle at each velocity field. The difference between the departure time and the arrival time is the travel time, which is marked as a blue line on the upper horizontal axis.

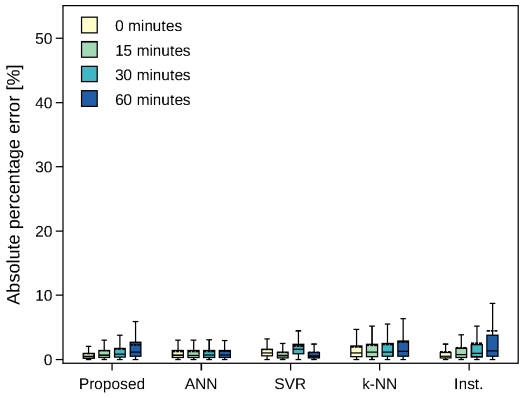
2.3.2 Determining hyper-parameters



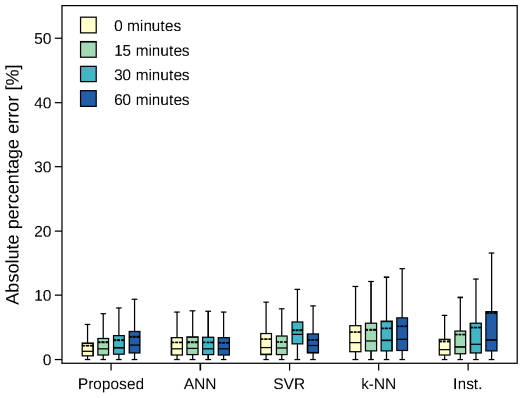
(a) Freeway I5-S: Peak period



(b) Freeway I210-E: Peak period



(c) Freeway I5-S: Off-peak period



(d) Freeway I210-E: Off-peak period

Figure 2.5: Absolute percentage errors (APE) of 5 different travel time forecasters with various horizons. Two examples of freeways in California, I5-S and I210-E, are studied during their peak periods (a) and (b); and their off-peak periods (c) and (d). Inside the box plots, the medians and mean values are marked as solid and dashed bars, respectively; different colors represent different prediction horizons.

Using Eq. (2.10), we have trained transition matrices with different pairs of the regularization parameter ρ and the forgetting factor λ. For each freeway, we have trained the transition matrix by all possible combinations of the following sets:

$$\begin{aligned} &\rho \in \{0, 0.1, 0.3, 1, 3, 10, 30, 100, 300, 1000, 3000, 10000\}, \\ &\lambda \in \{1, 0.999, 0.995, 0.99, 0.95\}. \end{aligned} \tag{2.24}$$

Table 2.1 shows the MAPE of travel time on the validation sets of the two freeways (peak periods only) by varying the hyper-parameters. The MAPE is not very sensitive to the regularization parameters, but it is influenced by the forgetting factors, as when the value of the forgetting factor is too low, this leads to a training of the transition matrices with not enough data. For

each case of the freeways, we have chosen the optimal pair among the tested parameter sets, which are:

$$\left(\rho, \lambda\right) = \begin{cases} (3000, 0.995) & \text{for I5-S} \\ (300, 1) & \text{for I210-E} \end{cases}. \tag{2.25}$$

We show that the optimal pairs of hyper-parameters chosen above work well for traffic prediction by showing an example of predicted velocity fields (Fig. 2.4). The prediction results (the contour plot) in Fig. 2.4 (c) and (d) show similar patterns to the ground truths (Fig. 2.4 (a) and (b)), which confirms that the chosen hyper-parameters are functioning well for predicting speeds and travel time.

2.3.3 Comparison of travel time with different forecasters

We examine the performance of our proposed method by comparing its performance with that of different prediction methods. We have chosen four various forecasters: the instantaneous travel time forecaster (abbreviated to *inst.*) as a real-time measurement-based method; the k-nearest neighbor (k-NN) as a historical data-based method; the support vector regression (SVR) and the vanilla artificial neural network (ANN) as representatives for direct methods. All the details of implementing these methods are explained in Appendix A.2.

Specifically, we evaluate travel time using these methods with different prediction horizons. We define a travel time at time t with a prediction horizon h-minutes as a travel time that a vehicle will experience when it departs h-minutes after the time t. For example, Fig. 2.4 (c) and (d) show travel time prediction with a 60-minute horizon at the current time of 2 PM. We assign four different values for h: 0, 15, 30, and 60 minutes.

Table 2.2: Mean absolute percentage error on test sets for each freeway

(a) Freeway I5-S

Prediction horizon	0 minutes		15 minutes		30 minutes		60 minutes	
	MAPE	Improvement rate	MAPE	Improvement rate	MAPE	Improvement rate	MAPE	Improvement rate
Proposed	3.07	**0.56**	4.15	**0.58**	5.15	0.60	7.03	0.59
ANN	4.68	0.33	4.64	0.54	4.62	**0.64**	4.92	**0.71**
SVR	7.59	-0.09	6.02	0.40	5.77	0.55	7.18	0.58
k-NN	6.14	0.12	6.89	0.31	7.63	0.41	9.00	0.48
Inst.	6.97	-	9.99	-	12.86	-	17.26	-

(b) Freeway I210-E

Prediction horizon	0 minutes		15 minutes		30 minutes		60 minutes	
	MAPE	Improvement rate	MAPE	Improvement rate	MAPE	Improvement rate	MAPE	Improvement rate
Proposed	5.08	**0.47**	5.76	**0.54**	6.49	0.57	8.09	0.60
ANN	6.21	0.35	6.25	0.50	6.48	**0.57**	6.37	**0.68**
SVR	9.62	-0.01	6.96	0.44	8.57	0.44	7.18	0.64
k-NN	9.07	0.05	9.49	0.24	9.77	0.36	10.29	0.49
Inst.	9.55	-	12.46	-	15.24	-	20.17	-

Figure 2.5 shows the average prediction errors (APE) of the results on the test sets. It shows that the proposed method always gives the best accuracy among others when $h = 0$ minute, for both freeways and in both peak and off-peak periods. For longer horizons, the performance of the proposed method is comparable to that of ANN and SVR, whereas it always performs better than k-NN and *inst.* in these results.

First of all, it is surprising that the proposed method has comparable errors with that of ANN and SVR for longer horizons. The ANN and SVR are direct methods, which means that they have a separate model for each horizon and each one has been trained independently. The proposed method, on the other hand, is an indirect method, which predicts the travel time of longer horizons based on previous predictions. In other words, it uses a model trained only once for all the horizons.

One could understand this from its superior performance at the 0-minute horizon. As it is seen in all the sub-figures of Fig. 2.5, the proposed method starts from a very small error, and then the error starts to increase gradually when extending the prediction horizon. This is simply due to the aggregate noise in Eq. (2.14). From the fact that covariance of the sum of two Gaussian random variables is always greater than the variance of each variable, the sum of the noise terms in Eq. (2.16) always produces larger covariance and therefore more substantial errors. However, since its initial 0-minute horizon error is very small compared to the other methods, the errors can remain relatively small even when the noise propagates and accumulates with time.

Compared to the other indirect methods, which are k-NN and instantaneous travel time forecaster, the proposed method shows better prediction regardless of traffic profiles and prediction horizons. We can find the reason by looking into the type of data that are considered in each method. The k-NN is highly dependent on historical data, whereas the instantaneous travel time forecaster uses only the real-time traffic measurement. Our prediction algorithm (Eq. (2.20)), on the other hand, utilizes both the real-time measurement ($\mathbf{v}_k$) and the historical information that is considered in the transition matrix ($\bar{H}_k$). This explains why it outperforms the other two methods.

Table 2.2 shows the mean average percentage errors (MAPE) in the peak periods and the corresponding *improvement rates*, which indicate how much the accuracy of travel time prediction is improved compared to that of the instantaneous travel time. For instance, in the case of Freeway I5-S during the peak periods, according to Table 2.2 (a), the proposed method improves the prediction accuracy compared to instantaneous travel time by 56% with the 0-minutes horizon. In contrast, ANN and SVR improve that by 33% and -9%, respectively.

Table 2.2 also confirms that the proposed method has the best prediction accuracy among all five forecasters for short horizons ($h = 0, 15$ min) and comparable performances to the best one for longer horizons ($h = 30, 60$ min). This result is promising since our approach has an additional degree of freedom to be used for arbitrary departure time and various starting points.

3 Generalization of Dynamic linear model

This chapter is written with its own notations, and the content of this chapter is available as a published paper:

> Kwak, Semin, Nikolas Geroliminis, and Pascal Frossard. "Traffic signal prediction on transportation networks using spatio-temporal correlations on graphs." IEEE Transactions on Signal and Information Processing over Networks 7 (2021): 648-659.

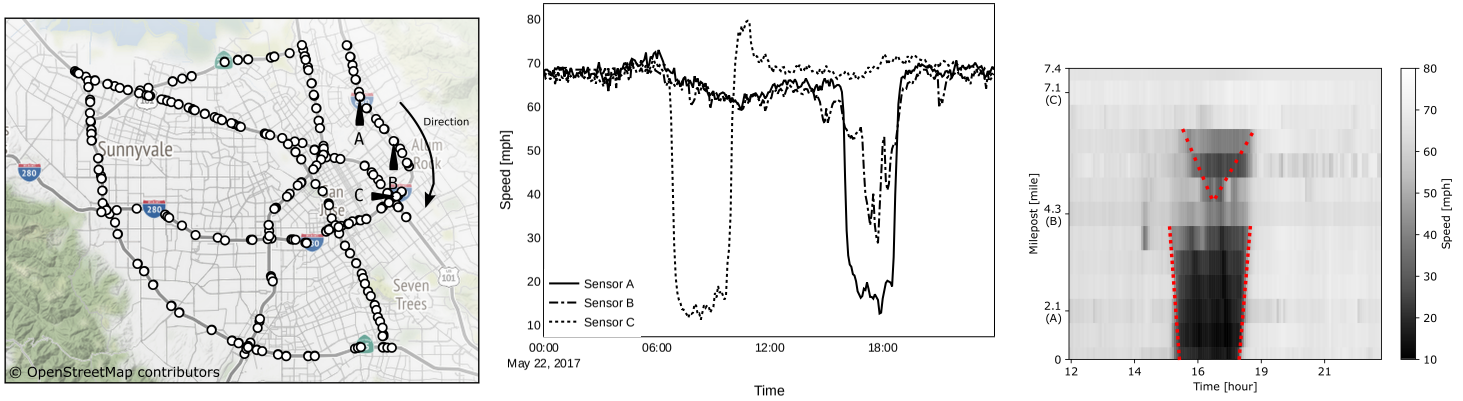
3.1 Introduction

Multivariate time-series prediction is an important task since many real-life problems can be modeled within this framework, such as weather forecasting [50]–[52], traffic prediction [3], [53]–[67], power consumption forecasting [58], [68], and others [54], [66], [69]–[71]. In transportation sensor networks, output signals from neighboring sensors may be similar or vastly different, as shown in Fig. 3.1(a) and (b). Therefore, in this example, sensor A's signal can be utilized to predict sensor B's as the two signals are well correlated. However, the signal of sensor C is not correlated with that of sensor B, so it may not contribute to the prediction; Sensor C is located after an intersection, and most traffic demands flow in another direction in the intersection, therefore, the sensor rarely suffers congestion. Naturally freeway congestion (expressed with a sharp decrease in the average speed of vehicles) is initiated at a bottleneck location such as an on-ramp merging area with high entrance flow or an incident location. Then, it propagates backwards with a finite speed, which is 3 to 4 times smaller than the speed of traffic. Fig. 3.1(c) shows an example of congestion propagation in I-280 and I-880 freeways in California. Note that there is a drastic decrease in the speed at a location (sensor B) and a time (around 3 pm) that propagates through the traffic stream (this is called a shockwave). Once demand for travel decreases congestion disappears by following the opposite trend during the offset of congestion with a forward moving wave. Note that this propagation speed

is not constant and depends on the concentration or density of vehicles (with units of veh/km) on the two sides of the shockwave. There are various theories in transportation science to describe the mechanisms of stop-and-go phenomena inspired by fluid and heat diffusion models (see [72] for an overview).

Due to complex spatio-temporal correlation, the choice of model greatly influences the predictive performance. For small-scale sensor networks, such correlations can be estimated directly from historical data [3], [55]–[58]. The vector Auto Regression (AR) is a representative model for multivariate time series forecasting [56]–[58]. In this model, regression parameters, or correlations between sensors, are estimated solely using historical data. In our previous work [56], we implemented a predictor that explicitly expresses the periodicity of traffic signals with temporally localized vector AR model. However, these data-driven models are not suitable for multivariate time series prediction with a large number of variables because the number of correlations to be estimated increases exponentially compared with the number of sensors, which causes incompleteness of the estimator (or overfitting).

Recently, many studies have prioritized the correlations among sensors by defining signals on graphs [59]–[67]. In particular, in transportation networks, the physical travel distance between sensors is a critical *a priori* information, the closer the sensors are in space, the higher the correlation [9]. Utilizing this information, the authors had extracted the signal's spatial features through the heat propagation kernel (or convolutional filter) and passed it to temporal blocks for forecasting, such as recurrent neural network (RNN) [59]–[62] and temporal convolutional layer (TCN) [63]–[66]. By introducing this prior information to complex deep neural networks, they achieved state-of-the-art performance in traffic prediction.



(a) Sensor locations of PEMS-BAY network. The distance between two consecutive sensors in a freeway is 0.6 mile in average.

(b) Signals on different sensors

(c) Speed profile for the evening peak over time and space. The day 2017-05-22 (Monday) is selected. The red dashed lines represent the waves that congestion propagates.

Figure 3.1: A transportation sensor network in California and signals of three different sensors on the network. Although the sensors B and C are close to each other in distance, two traffic signals from these sensors show very different patterns.

However, the two predictors (with and without graphs) each have their own drawbacks. In the former case, to the best of our knowledge, all studies, which currently show the best performance, construct predictors based on deep neural networks. Therefore, these models require expensive tuning processes of many hyperparameters and relatively long training due to numerical optimization processes. In the latter case, on the other hand, it can be inefficient concerning the prediction accuracy, especially for large networks when the structural information becomes important.

This chapter proposes a new model that combines the advantages of different frameworks by implanting the sensors' structural information into the existing data-driven model [56], inheriting the periodicity modeling for the traffic signal. In most studies, the periodicity of the traffic signal is taken as the input feature of the predictor, such as an encoded vector that represents the time of the day or the day of the week, but the study [56] instead induces the periodicity of the signal more clearly by making the model itself different for each time. Each model has a matrix, which should be estimated by historical data, representing the correlation between signals at two consecutive time intervals. As the size of the network is proportional to the size of the matrix, a larger network can lead to overfitting. In this chapter, we resolve the overfitting problem by approximating this matrix to the one derived from data-independent graph topological information, therefore, we estimate only the remainder by data. In detail, we transform the graph topological information into heat diffusion kernels, which is introduced in [73], and approximate the matrix to a combination of the heat diffusion kernels. In the process, we introduce some hyper-parameters. For example, one determines which of the prior or historical datasets is more reliable. Most of the existing studies estimate hyper-parameters through exhaustive search as a cross-validation method using a validation set, but we estimate hyper-parameters directly from data by utilizing Bayesian inference [5]. As a result, the estimation process is relatively fast as most parameter estimation is performed by analytic calculations except a few ones requiring a numerical optimization process. Besides, our model is strongly interpretable. For example, through the hyper-parameter, it can be seen that during the peak period, traffic prediction is relatively more dependent on data than structural information compared to the non-peak period. Also, most importantly, predictors based on this model showed comparable performance with a much shorter learning time than state-of-the-art models. Especially, the proposed model shows great long-term prediction performance as the model captures well the periodicity of traffic signals. Since the proposed model requires a minimal number of hyper-parameter tuning, it might be applied to other daily periodic graph signal prediction problems easily (e.g., weather forecasting, daily energy consumption prediction). Here we summarize contributions of the work:

- We propose a novel traffic prediction method that successfully integrate graph structural information to the existing data-driven model [56]. Hyper-parameters are learned directly from data through Bayesian inference rather than by exhaustive search.
- Therefore, the training time required for inference is minimal. The trained model is straightforward to analyze, unlike other deep neural network-based models.

Table 3.1: The notations and definitions used in this chapter.

$\mathscr{R}^m$	m-dimensional Euclidean space
$a, \mathbf{a}, \mathbf{A}$	Scalar, vector, matrix
diag($\mathbf{a}$)	The diagonal matrix whose diagonal elements are from the vector $\mathbf{a}$
diag($\mathbf{A}$)	The vector whose elements are the diagonal components of the matrix $\mathbf{A}$
$\mathbf{I}$	Identity matrix
$\mathbf{1}$	All one vector
$e^{\mathbf{A}}$	$\lim_{n\to\infty}\left(\mathbf{I}+\frac{1}{n}\mathbf{A}\right)^n = \sum_{n=0}^{\infty}\frac{1}{n!}\mathbf{A}^n$
$[\mathbf{A}]_{i,j}$	The element of i-th row and j-th column of the matrix $\mathbf{A}$
$[\mathbf{A}]_{i,:}$	The slice of i-th row of the matrix $\mathbf{A}$
$\lvert\mathbf{A}\rvert$	The determinant of the matrix $\mathbf{A}$
$\lvert\mathscr{S}\rvert$	The cardinality of the set $\mathscr{S}$
$\mathcal{N}(\mu, \sigma^2)$	A Gaussian distribution which has the probability density function $f(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)$
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	A multivariate Gaussian distribution which has the probability density function $f(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^N\lvert\boldsymbol{\Sigma}\rvert}}\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$
$\mathcal{N}(\mathbf{M}, \sigma^2)$	$\prod_{i,j}\mathcal{N}([\mathbf{M}]_{i,j}, \sigma^2)$
$\mathcal{N}(\mathbf{M}, \boldsymbol{\Sigma})$	$\prod_{i}\mathcal{N}([\mathbf{M}]_{i,:}, \boldsymbol{\Sigma})$

- It shows prediction performance comparable with deep learning methods especially for long-term prediction.

3.2 Data model

In this section, we describe a mathematical model that represents a relationship between traffic signals that are different in time. First, we define traffic signals on a graph and introduce an existing prediction model [56] using this signals. Then, we suggest a model extending the previous one that is applicable for large scale networks by exploiting graph information.

Mathematical notations that are used in this chapter are in Table 3.1.

3.2.1 Graph signal

We start with modeling a transportation network using a graph. We define an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$; $\mathcal{V}$ is a set of nodes where each $v \in \mathcal{V}$ denotes a node (sensor) on the graph; $\mathcal{E}$ is a set of edges where each of the edges connects two nodes. We define a signal on the nodes of the graph with a traffic feature, in this chapter, for instance, speed, which is expressed as a vector $\mathbf{x}_t^d \in \mathcal{R}^N$ of a day d and time t, where the constant N is the number of nodes. Therefore, the vector $\mathbf{x}_t^d$ represents a snapshot of speeds at a particular time and day. Especially, we express the day index on the vector representation to exploit the periodicity of traffic signals later.

3.2.2 Dynamic linear model (DLM)

In our previous study [56], we defined a state equation of traffic in a small-scale transportation network (a path graph) as temporally localized linear models as follows:

$$\mathbf{x}_{t+1}^d = \mathbf{H}_t \mathbf{x}_t^d + \mathbf{n}_t^d, \forall\, t \in [0, T-1]\,. \tag{3.1}$$

We called this model the Dynamic linear model (DLM). The first time index ($t = 0$) corresponds to the beginning of a day (midnight in our work), and the last index ($t = T-1$) refers to the end of the day. Each entry of the noise vector $\mathbf{n}_t^d \in \mathcal{R}^N$ is assumed to be an independent and identically distributed (i.i.d.) random variable, which follows a Gaussian distribution $\mathcal{N}(0, \alpha_t^{-1})$. Here the precision parameter α_t explains how precisely a data pair $(\mathbf{x}_t^d, \mathbf{x}_{t+1}^d)$ fits to the model. The transition matrix $\mathbf{H}_t$ represents the linear relationship between traffic signals $\mathbf{x}_t^d$ and $\mathbf{x}_{t+1}^d$.

The most important motivation behind this model is that the propagation of traffic features over time occurs periodically on a daily basis. Consequently, we modeled that the transition matrix $\mathbf{H}_t$ as a time-variant matrix that contains temporally localized (only between two consecutive traffic features) spatio-temporal correlations of every sensor pair regardless of the day of the week, noting that the transition matrix does not have the day index. In other words, we assumed the correlations are identical both for weekends and weekdays [56].

In the work [56], the transition matrix is estimated by maximizing the likelihood (note that we ignore some parameters such as the regularization parameter and the forgetting factor introduced in the work for the brevity) as follows:

$$\bar{\mathbf{H}}_t = \underset{\mathbf{H}_t}{\operatorname{argmax}}\, f(\mathbf{X}_{t+1} | \mathbf{X}_t, \mathbf{H}_t, \alpha_t) = \mathbf{X}_{t+1}\mathbf{X}_t^T(\mathbf{X}_t\mathbf{X}_t^T)^{-1}, \tag{3.2}$$

where the collection of the m-past signals $\mathbf{X}_t = \begin{pmatrix} \mathbf{x}_t^0 & \mathbf{x}_t^1 & \cdots & \mathbf{x}_t^{m-1} \end{pmatrix}$. Therefore, the optimal transition matrix is solely determined by the historical data $\mathbf{X}_t$ and $\mathbf{X}_{t+1}$. From Eq. (3.2) we see that the matrix $\mathbf{X}_t\mathbf{X}_t^T$ can be an ill-conditioned matrix when N is large. In other words, the transition matrix $\bar{\mathbf{H}}_t$ can be overfitted by data. In the following subsection, we suggest a method to avoid this problem by utilizing graph topological information.

3.2.3 DLM with graph topological information

In this subsection, we suggest a way to avoid the overfitting problem approximating the transition matrix to a heat diffusion matrix. To achieve this goal, we first define a weight matrix that contains all edge weights between node v_i and v_j using a Gaussian kernel weighting function with a threshold constant κ:

$$[\mathbf{W}]_{i,j} = \begin{cases} e^{-\frac{\text{dist}^2(i,j)}{\sigma^2}}, & \text{if dist}(i,j) \leq \kappa \\ 0, & \text{otherwise.} \end{cases} \tag{3.3}$$

The function dist(i, j) denotes the shortest travel distance on $\mathcal{G}$ between the node v_i and v_j:

$$\text{dist}(i,j) = \min\{\text{dist}(v_i \to v_j), \text{dist}(v_j \to v_i)\}, \tag{3.4}$$

where the function dist$(v_i \to v_j)$ represents the shortest travel distance from node v_i to node v_j. As the graph $\mathcal{G}$ is undirected, the weight matrix is a symmetric matrix, i.e., $\mathbf{W}^T = \mathbf{W}$.

The constants σ and κ are the kernel width and the distance threshold. If the kernel width is large, the correlation of a pair of nodes is strong (close to one) even though the shortest travel distance between the two nodes is large. On the other hand, the smaller the threshold is, the sparser the weight matrix is.

The graph heat diffusion model [73] explains how each vertex propagates its heat to its neighbors on the graph over time. As congestion evolves from one location to its neighbor over time, we can express the change of traffic features by the heat diffusion model, especially for short-term traffic changes since the total traffic volume of a network is well preserved for the short-term in general.

The kernel on graphs that supports the heat diffusion model is introduced by [73]:

$$\mathbf{H}^{\mathcal{G}}(\tau) = e^{-\tau \mathbf{L}(\mathcal{G})}, \tag{3.5}$$

where the constant τ denotes the diffusion period and the matrix $\mathbf{L}(\mathcal{G})$ is the Laplacian of a graph $\mathcal{G}$. The matrix is defined as

$$\mathbf{L}(\mathcal{G}) = \text{diag}(\mathbf{W}\mathbf{1}) - \mathbf{W}. \tag{3.6}$$

By definition, two extreme heat diffusion kernels of a connected graph $\mathcal{G}$ are:

$$\mathbf{H}^{\mathcal{G}}(\tau) = \begin{cases} \mathbf{I}, & \text{when } \tau \to 0, \\ \frac{1}{N}\mathbf{1}\mathbf{1}^T, & \text{when } \tau \to \infty, \end{cases} \tag{3.7}$$

where $\mathbf{1}$ is the vector whose elements are all one.

Therefore, with the heat diffusion kernel, we can describe the diffusion of a traffic signal

through the graph $\mathcal{G}$ as follows:

$$\tilde{\mathbf{x}}_{t+1}^{d}(\tau) = \mathbf{H}^{\mathcal{G}}(\tau)\mathbf{x}_{t}^{d}. \tag{3.8}$$

We call the vector $\tilde{\mathbf{x}}_{t+1}^{d}(\tau)$ the internally diffused signals from $\mathbf{x}_{t}^{d}$ by the diffusion period τ on the graph $\mathcal{G}$ over one incremental time step.

Here, we define a convex combination of the heat diffusion kernels of K different predetermined diffusion periods with a set $\mathcal{T} = \{\tau^{(0)}, \tau^{(1)}, \cdots, \tau^{(K-1)}\}$[1] as

$$\mathbf{H}^{\mathcal{G}}(\mathcal{T}) = \sum_{\tau \in \mathcal{T}} \pi^{(\tau)} \mathbf{H}^{\mathcal{G}}(\tau), \tag{3.9}$$

where $\sum_{\tau \in \mathcal{T}} \pi^{(\tau)} = 1$. The mixture retains the property that the total input volume is preserved through the diffusion process as shown in Appendix B.1, i.e., $\mathbf{1}^{T}\mathbf{H}^{\mathcal{G}}(\mathcal{T})\mathbf{x}_{t}^{d} = \mathbf{1}^{T}\mathbf{x}_{t}^{d}$.

We embed heat diffusion kernels into DLM to exploit topological information of the transportation network. The key idea is to express the transition matrix as a small variant from a mixture of diffusion kernels. We decompose the transition matrix into the time-variant internal diffusion and residual as follows:

$$\mathbf{H}_{t} = \mathbf{H}_{t}^{\mathcal{G}}(\mathcal{T}) + \text{residual} \tag{3.10}$$

so that the internal diffusion matrix $\mathbf{H}_{t}^{\mathcal{G}}(\mathcal{T})$ preserves the total traffic volume over time, i.e., $\mathbf{1}^{T}\mathbf{H}_{t}^{\mathcal{G}}(\mathcal{T})\mathbf{x}_{t}^{d} = \mathbf{1}^{T}\mathbf{x}_{t}^{d}$. Here, the time dependent internal transition matrix can be safely defined as in Eq. (3.9) by substituting the time-invariant parameter $\pi^{(\tau)}$ for the time-variant one $\pi_{t}^{(\tau)}$ because of the volume conservation property. The internal diffusion matrix represents how the current signal $\mathbf{x}_{t}^{d}$ diffuses through the transportation network (endogenous) whereas the residual represents how much the traffic situation is getting better or worse in the next time step based on the current signal (exogenous).

With this interpretation, we model the prior distribution of the transition matrix as:

$$f(\mathbf{H}_{t}|\gamma_{t}, \Pi_{t}, \mathcal{G}) = \mathcal{N}\left(\mathbf{H}_{t}^{\mathcal{G}}(\mathcal{T}), \gamma_{t}^{-1}\right), \tag{3.11}$$

where the precision parameter γ_{t} represents how precisely the diffusion matrix explains the transition matrix and $\Pi_{t} = \{\pi_{t}^{(\tau)} | \tau \in \mathcal{T}\}$.

The decomposition allows us to utilize data more efficiently during the estimation process later. In Eq. (3.1), the transition matrix is a variable to be estimated from the data. Since the dimension of this matrix is N^2, an increase in the number of sensors causes the estimation

[1] We predetermine the set $\mathcal{T}$ with two diffusion periods τ_0 and τ_∞ that correspond to each extreme case in Eq. (3.7), respectively. In practice, we set τ_0 as the biggest one that satisfies $\left\|\mathbf{H}^{\mathcal{G}}(\tau) - \mathbf{I}\right\|_2 < \epsilon$ and τ_∞ as the smallest one that satisfies $\left\|\mathbf{H}^{\mathcal{G}}(\tau) - 1/N\mathbf{1}\mathbf{1}^{T}\right\|_2 < \epsilon$ with a predefined set $\tau \in$ linspace(-10,10,0.1), where the set contains evenly spaced (0.1) numbers from -10 to 10. After that, we define $\mathcal{T} =$ logspace(τ_0, τ_∞, K), where the function returns K evenly spaced numbers on a log scale from τ_0 to τ_∞.

Algorithm 2 Inference of parameters

function INFERENCE($\mathbf{W}, K, \mathbf{X}_{1:T}$)
  Set $\mathscr{T} = \text{logspace}(\tau_0, \tau_\infty, K)$
  Define $\mathbf{L}(\mathscr{G})$ by Eq. (3.6)
  Define the function $\mathbf{H}^{\mathscr{G}}(\tau) = e^{-\tau\mathbf{L}(\mathscr{G})}$
  for $t \in [0, T-2]$ **do**
    Infer $\hat{\alpha}_t$, $\hat{\gamma}_t$ and $\hat{\Pi}_t$ by solving (3.25)
    Infer $\hat{\mathbf{H}}_t$ by Eq. (3.19)
  end for
return $\hat{\mathbf{H}}_t, \forall t$
end function

of more elements, which results in an overfitting problem. This is the biggest impediment to extending DLM to large networks. Still, if the structural information is set as *a priori* through Eq. (3.11), the problem can be effectively avoided even if the number of sensors increases. Assuming the graph $\mathscr{G}$ and the period set $\mathscr{T}$ are predefined, the internal diffusion matrix only depends on the parameters $\pi_t^{(\tau)}$. By setting the number of diffusion periods to be much smaller than that of sensors i.e., $|\mathscr{T}| \ll N$, we can describe the major part of the transition matrix by the internal diffusion matrix with a few parameters when the sampling interval (the time difference of two consecutive time indices) is relatively short, with likely preservation of the traffic volumes, i.e., $\mathbf{1}^T\mathbf{x}_{t+1}^d \approx \mathbf{1}^T\mathbf{x}_t^d$. Consequently, we only need to exploit data to infer the parameters $\pi_t^{(\tau)}$ and the residual part whose norm is small with the decomposition.

3.3 Prediction and inference

This section describes how to estimate modeling parameters and predict graph signals by using the model. Both the estimation and the prediction were performed by maximizing the posterior distribution of each variable. Especially for hyperparameters, we utilize Bayesian inference to estimate them instead of exhaustive search.

3.3.1 Inference of the transition matrix

We infer the transition matrix by maximizing its posterior distribution:

$$\hat{\mathbf{H}}_t = \underset{\mathbf{H}_t}{\text{argmax}}\ f(\mathbf{H}_t|\mathbf{X}_t, \mathbf{X}_{t+1}, \alpha_t, \gamma_t, \Pi_t, \mathscr{G}), \tag{3.12}$$

which is proportional to the product of the prior and the likelihood by Bayes' rule:

$$\text{Posterior dist.} \propto f(\mathbf{H}_t|\gamma_t, \Pi_t, \mathscr{G}) f(\mathbf{X}_{t+1}|\mathbf{X}_t, \mathbf{H}_t, \alpha_t). \tag{3.13}$$

Maximizing the posterior distribution can be interpreted as balancing between the prior and likelihood of the transition matrix. For example, if there is no topological information about

sensors, the transition matrix should be inferred by considering the training dataset only. In this case, we can set the prior distribution as a uniform distribution, meaning that there is no strong preference for a particular value of the transition matrix; the most probable transition matrix becomes the maximum likelihood solution, which is Eq. (3.2):

$$\begin{aligned}\hat{\mathbf{H}}_t|\text{No topological info.} &:= \bar{\mathbf{H}}_t \\ &= \underset{\mathbf{H}_t}{\operatorname{argmax}}\, f(\mathbf{X}_{t+1}|\mathbf{X}_t, \mathbf{H}_t, \alpha_t) \\ &= \mathbf{X}_{t+1}\mathbf{X}_t^T(\mathbf{X}_t\mathbf{X}_t^T)^{-1}.\end{aligned} \tag{3.14}$$

On the other hand, if we do not have any measurements, the most probable transition matrix should be the maximizer of the prior distribution:

$$\hat{\mathbf{H}}_t|\text{No measurements} = \underset{\mathbf{H}_t}{\operatorname{argmax}}\, f(\mathbf{H}_t|\gamma_t, \Pi_t, \mathscr{G}) = \mathbf{H}_t^{\mathscr{G}}(\mathscr{T}). \tag{3.15}$$

Since we use both prior and data measurements, the actual optimal transition matrix becomes a combination of these two. According to the dynamic linear model, the likelihood

$$\begin{aligned}&f(\mathbf{X}_{t+1}|\mathbf{H}_t, \mathbf{X}_t, \alpha_t) \\ &\qquad \propto e^{-\frac{1}{2}\mathrm{tr}\{\alpha_t(\mathbf{X}_{t+1}-\mathbf{H}_t\mathbf{X}_t)(\mathbf{X}_{t+1}-\mathbf{H}_t\mathbf{X}_t)^T\}}\end{aligned} \tag{3.16}$$

and the prior

$$f(\mathbf{H}_t|\gamma_t, \Pi_t, \mathscr{G}) \propto e^{-\frac{1}{2}\mathrm{tr}\{\gamma_t(\mathbf{H}_t-\mathbf{H}_t^{\mathscr{G}}(\mathscr{T}))(\mathbf{H}_t-\mathbf{H}_t^{\mathscr{G}}(\mathscr{T}))^T\}}. \tag{3.17}$$

Therefore, by Eq. (3.13), (3.16) and (3.17),

$$\begin{aligned}&f(\mathbf{H}_t|\mathbf{X}_{t+1}, \mathbf{X}_t, \alpha_t, \gamma_t, \Pi_t, \mathscr{G}) \\ &\quad \propto e^{-\frac{1}{2}\alpha_t\mathrm{tr}\{(\mathbf{X}_{t+1}-\mathbf{H}_t\mathbf{X}_t)(\mathbf{X}_{t+1}-\mathbf{H}_t\mathbf{X}_t)^T\}} \\ &\qquad \cdot e^{-\frac{1}{2}\gamma_t\mathrm{tr}\{(\mathbf{H}_t-\mathbf{H}_t^{\mathscr{G}}(\mathscr{T}))(\mathbf{H}_t-\mathbf{H}_t^{\mathscr{G}}(\mathscr{T}))^T\}} \\ &\quad \propto e^{-\frac{1}{2}\mathrm{tr}\{(\mathbf{H}_t-\hat{\mathbf{H}}_t)(\alpha_t\mathbf{X}_t\mathbf{X}_t^T+\gamma_t I)(\mathbf{H}_t-\hat{\mathbf{H}}_t)^T\}},\end{aligned} \tag{3.18}$$

where

$$\begin{aligned}\hat{\mathbf{H}}_t &= (\bar{\mathbf{H}}_t\alpha_t\mathbf{U}_t\Lambda_t + \mathbf{H}_t^{\mathscr{G}}(\mathscr{T})\gamma_t\mathbf{U}_t)(\alpha_t\Lambda_t + \gamma_t\mathbf{I})^{-1}\mathbf{U}_t^T \\ &= \bar{\mathbf{H}}_t\alpha_t\mathbf{U}_t\Lambda_t(\alpha_t\Lambda_t + \gamma_t\mathbf{I})^{-1}\mathbf{U}_t^T \\ &\qquad + \mathbf{H}_t^{\mathscr{G}}(\mathscr{T})\gamma_t\mathbf{U}_t(\alpha_t\Lambda_t + \gamma_t\mathbf{I})^{-1}\mathbf{U}_t^T,\end{aligned} \tag{3.19}$$

with the eigendecomposition of $\mathbf{X}_t\mathbf{X}_t^T = \mathbf{U}_t\Lambda_t\mathbf{U}_t^T$. Therefore, $f(\mathbf{H}_t|\mathbf{X}_{t+1}, \mathbf{X}_t, \alpha_t, \gamma_t, \Pi_t, \mathscr{G})$ is a multivariate Gaussian distribution with mean $\hat{\mathbf{H}}_t$ and the covariance of each row; $(\alpha_t\mathbf{X}_t\mathbf{X}_t^T + \gamma_t I)^{-1}$.

Here, we measure how much each part contributes to the transition matrix

$$c_t^{\text{data}} = \frac{w_t^{\text{data}}}{w_t^{\text{data}} + w_t^{\text{prior}}},\ c_t^{\text{prior}} = \frac{w_t^{\text{prior}}}{w_t^{\text{data}} + w_t^{\text{prior}}} \tag{3.20}$$

by defining the weight of each part

$$\begin{aligned} w_t^{\text{data}} &= \left\| \alpha_t \mathbf{U}_t \Lambda_t (\alpha_t \Lambda_t + \gamma_t \mathbf{I})^{-1} \mathbf{U}_t^T \right\|_F, \\ w_t^{\text{prior}} &= \left\| \gamma_t \mathbf{U}_t (\alpha_t \Lambda_t + \gamma_t \mathbf{I})^{-1} \mathbf{U}_t^T \right\|_F. \end{aligned} \tag{3.21}$$

Note that these weights depend on the precision parameters α_t and γ_t. If the data precision parameter α_t is relatively large compared to γ_t, then $c_t^{\text{data}} > c^{\text{prior}}$, meaning that the contribution of data measurements is larger than that of the prior information.

3.3.2 Inference of other parameters

For the next step, we infer parameters α_t, γ_t, and Π_t. Similar to inferring the most probable transition matrix, we infer the most probable α_t, γ_t, and Π_t by maximizing the following posterior distribution:

$$\hat{\alpha}_t, \hat{\gamma}_t, \hat{\Pi}_t = \underset{\alpha_t, \gamma_t, \Pi_t}{\text{argmax}}\ f(\alpha_t, \gamma_t, \Pi_t | \mathbf{X}_{t+1}, \mathbf{X}_t). \tag{3.22}$$

Setting the prior distribution $f(\alpha_t, \gamma_t, \Pi_t)$ as a uniform distribution based on the assumption that there is no preference for a certain value for these parameters before inferring, the objective changes to maximize *evidence* $f(\mathbf{X}_{t+1} | \mathbf{X}_t, \alpha_t, \gamma_t, \Pi_t)$ [5] since

$$\begin{aligned} f(\alpha_t, \gamma_t, \Pi_t | \mathbf{X}_{t+1}, \mathbf{X}_t) &\propto f(\mathbf{X}_{t+1} | \mathbf{X}_t, \alpha_t, \gamma_t, \Pi_t) f(\alpha_t, \gamma_t, \Pi_t) \\ &\propto f(\mathbf{X}_{t+1} | \mathbf{X}_t, \alpha_t, \gamma_t, \Pi_t). \end{aligned} \tag{3.23}$$

In Appendix B.2, we show that the evidence is

$$\begin{aligned} &f(\mathbf{X}_{t+1} | \mathbf{X}_t, \alpha_t, \gamma_t, \Pi_t) \\ &\quad = \int f(\mathbf{X}_{t+1} | \mathbf{X}_t, \mathbf{H}_t, \alpha_t) f(\mathbf{H}_t | \gamma_t, \Pi_t) d\mathbf{H}_t \\ &\quad = \mathcal{N}(\mathbf{H}_t^{\mathcal{G}}(\mathcal{T}) \mathbf{X}_t, \alpha_t^{-1} \mathbf{I} + \gamma_t^{-1} \mathbf{X}_t^T \mathbf{X}_t). \end{aligned} \tag{3.24}$$

Therefore, we infer the most probable hyper-parameters by maximizing the log-evidence with a quasi-newton method (L-BFGS-B [74]):

$$\begin{aligned} \underset{\alpha_t, \gamma_t, \Pi_t}{\text{maximize}} \quad & \log \mathcal{N}(\mathbf{H}_t^{\mathcal{G}}(\mathcal{T}) \mathbf{X}_t, \alpha_t^{-1} \mathbf{I} + \gamma_t^{-1} \mathbf{X}_t^T \mathbf{X}_t) \\ \text{subject to} \quad & 0 \le \pi_t^{(\tau)} \le 1\ \forall \tau \in \mathcal{T},\ 0 < \alpha_t,\ 0 < \gamma_t, \\ & \sum_{\tau \in \mathcal{T}} \pi_t^{(\tau)} = 1. \end{aligned} \tag{3.25}$$

Algorithm 3 Prediction of traffic features (h-steps ahead)

function PREDICTION($\mathbf{x}_t^d, h, \hat{\mathbf{H}}_t, \cdots, \hat{\mathbf{H}}_{t+h-1}$)
  Set $\mathbf{p} = \mathbf{x}_t^d$
  for $i \in [0, h-1]$ **do**
    Set $\mathbf{p} = \hat{\mathbf{H}}_{t+i}\mathbf{p}$
  end for
  $\mathbf{x}_{t+h|t} = \mathbf{p}$
return $\mathbf{x}_{t+h|t}$
end function

Algorithm 2 summarizes the inference processes.

We emphasize that parameter inference through evidence maximization prevents overfitting of the transition matrix to either data measurements or prior information. In Eq. (3.24) we calculate the evidence by marginalizing the transition matrix. In other words, we set the transition matrix as a *random variable* instead of fixing it as a representative value, e.g., maximum likelihood estimator. Noting that these parameters determine the contributions of measurements and priors when the transition matrix is estimated in Eq. (3.19), the marginalization process automatically penalizes the transition matrix to avoid the extreme cases [5].

3.3.3 Prediction of traffic features

Prediction of traffic features is performed by extracting and exploiting as much information as possible from measurements and prior knowledge. Mathematically, we can express a traffic signal that we want to predict as a random variable since the signal defined in the future is entirely unknown. In this chapter, therefore, we try to infer the probability density function of the signal $\mathbf{x}_{t+h}^d$

$$f(\mathbf{x}_{t+h}^d | \mathbf{x}_t^d, \mathbf{x}_{t-1}^d \cdots, \mathcal{G}), \tag{3.26}$$

where the time indices t and $t+h$ represent respectively the current time and the future time index (h-steps ahead) that we want to predict. In the expression, the probability density function is conditioned by the signals $\{\mathbf{x}_t^d, \mathbf{x}_{t-1}^d, \cdots\}$ and the graph $\mathcal{G}$ that represents a set of measurements and prior structural information, respectively.

In reality, it is common to limit the number of measurements to a fixed-sized one in a training set. In addition to the training set that contains measurements apart from the day to be predicted, it is crucial to keep measurements just before t, as the temporal correlation is strong when the time difference is small. As a result, we estimate the density function that is conditioned by a training set, the p-most recent measurements, and the graph $\mathcal{G}$:

$$f(\mathbf{x}_{t+h}^d | \mathbf{x}_t^d, \mathbf{x}_{t-1}^d, \cdots, \mathbf{x}_{t-(p-1)}^d, \mathbf{X}_{0:T-1}, \mathcal{G}), \tag{3.27}$$

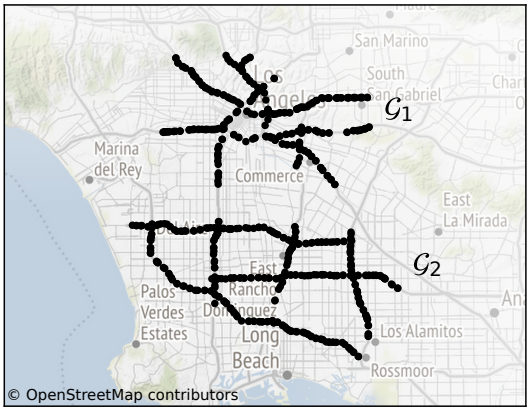


Figure 3.2: Transportation sensor networks (District 7 area in California) that are used for evaluating the proposed method.

where the training set $\mathbf{X}_{0:T-1}$ contains signals from $t = 0$ to $t = T-1$ of multiple days $d \in [0, m-1]$. The dynamic linear model further simplifies the distribution (3.27) as follows

$$f(\mathbf{x}_{t+h}^d | \mathbf{x}_t^d, \mathbf{X}_{t:t+h}, \mathcal{G}) \tag{3.28}$$

because of the temporal locality of the model.

We define a predictor $\mathbf{x}_{t+h|t}^d$ at the time step t for the horizon h as the maximizer of the probability density function

$$\mathbf{x}_{t+h|t}^d := \underset{\mathbf{x}_{t+h}^d}{\operatorname{argmax}} f(\mathbf{x}_{t+h}^d | \mathbf{x}_t^d, \mathbf{X}_{t:t+h}, \mathcal{G}). \tag{3.29}$$

In other words, we define the predictor $\mathbf{x}_{t+h|t}^d$ as the most probable $\mathbf{x}_{t+h}^d$ based on the current measurement vector $\mathbf{x}_t^d$, the training set $\mathbf{X}_{t:t+h}$, and the graph $\mathcal{G}$.

Proposition 1. *The posterior distribution $f(\mathbf{x}_{t+h}^d | \mathbf{x}_t^d, \mathbf{X}_{t:t+h}, \mathcal{G})$ is a Gaussian distribution that has the mean vector $\hat{\mathbf{H}}_{t+h-1} \cdots \hat{\mathbf{H}}_t \mathbf{x}_t^d$ assuming $f(\mathbf{H}_t | \mathbf{X}_t, \mathbf{X}_{t+1}, \alpha_t, \gamma_t, \Pi_t, \mathcal{G}) = \delta(\mathbf{H}_t - \hat{\mathbf{H}}_t)$, where the Dirac delta function $\delta(x) = 1$ when $x = 0$ and $\delta(x) = 0$, otherwise. The most probable transition $\hat{\mathbf{H}}_t$ is the maximizer of the posterior distribution $f(\mathbf{H}_t | \cdot)$.*

Proof. See Appendix B.3. □

Since the mean value of a Gaussian distribution maximizes the distribution, the optimal predictor is

$$\mathbf{x}_{t+h|t}^d = \hat{\mathbf{H}}_{t+h-1} \cdots \hat{\mathbf{H}}_t \mathbf{x}_t := \hat{\mathbf{H}}_{t+h-1 \leftarrow t} \mathbf{x}_t^d. \tag{3.30}$$

Therefore, the most probable signal $\mathbf{x}_{t+h}^d$ is the successive propagation of the current mea-

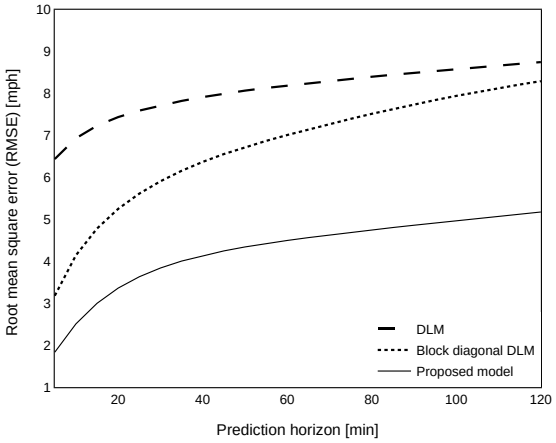


Figure 3.3: Prediction accuracy (RMSE) for the three different models on the PEMS-BAY dataset. Each model represents respectively a single DLM (without topological information), separate multiple DLMs for each freeway, and the proposed model (a DLM with topological information).

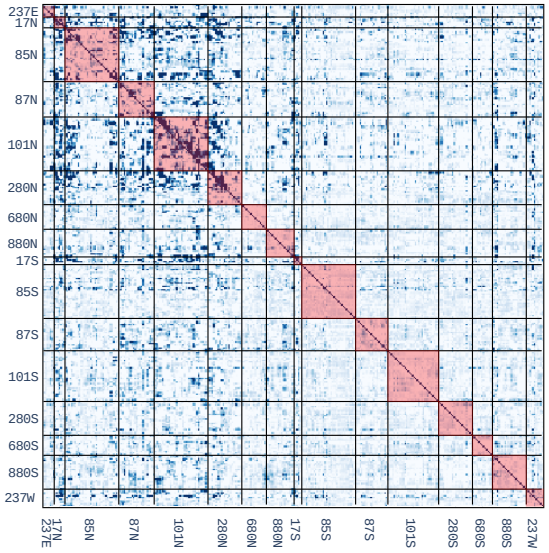


Figure 3.4: The heatmap of the elements in an estimated transition matrix $\mathbf{H}_t$ of the proposed model. Darker colors represent larger absolute values. The sensors are grouped by freeways and ordered from upstream to downstream within each freeway. Each axis shows the name of the freeways. The sensors' correlations within the same freeway are represented as red-shaded areas (block-diagonal elements of the matrix). The separate multiple DLMs only use block diagonal elements in the matrix.

surement vector $\mathbf{x}_t^d$ through the most probable transition matrices that coincides with a straightforward computation with Eq. (3.1) ignoring the noise term. Therefore, the prediction for any horizon is just a matrix multiplication. Algorithm 3 summarizes this.

3.4 Experiments

3.4.1 Settings

The proposed method was evaluated on different transportation networks. Figure 3.2 shows the networks ($\mathcal{G}_1$ and $\mathcal{G}_2$) consisting of respectively 288 and 357 sensors with multiple freeways that are connected through ramps. They experience significant levels of congestion in the morning and evening peaks at various locations. These networks connect many origins and destinations with complex demand profiles, creating propagation of congestion that is different in duration, size, and time of occurrence. The PEMS-BAY dataset was also used as a benchmark to compare with other state-of-the-art models [59], [65]. This data set consists of data measured from 325 sensors (Fig. 3.1(a)) on the freeways of San Francisco Bay area. The training and test dataset were constructed in the same way as [59], [65] to achieve a fair comparison.

The sampling interval of each dataset is 5 minutes by default, and in the following subsection, it is downsampled to 10 and 15 minutes, respectively, for a specific experiment. Both datasets of networks $\mathcal{G}_1$ and $\mathcal{G}_2$ contain 209 days of speed data, and each of those is divided into a training set and a test set at an 8:2 ratio by default. Another ratio is applied in Section 3.4.2 for a specific experiment.

We used the root mean square error (RMSE) as an error metric to measure the accuracy of prediction since the solution in Eq. (3.30) is also the optimal under the minimum mean squares error (MMSE) sense [56]. The RMSE of a method with the prediction horizon h is defined as

$$\text{RMSE}(h, \text{method}) = \sqrt{\text{mean}(\mathbf{x}_{t+h|t}^{\text{method}} - \mathbf{x}_{t+h})^2}, \tag{3.31}$$

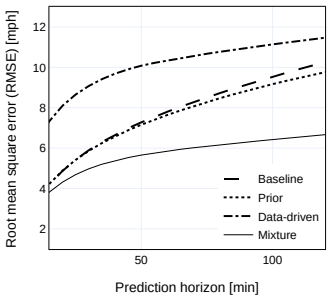
where the mean value is evaluated over all t in the test set.

For prediction horizons, we set from 5 minutes to 120 minutes every 5 minutes. In our previous work [56], on a freeway with a total length of about 60 miles (similar to the longest path of the networks considered here), the actual travel time is about 70 minutes under usual congestion. In the most severe congestion, the maximum travel time is about 100 minutes, and accordingly, we set the maximum prediction horizon to 120 minutes.

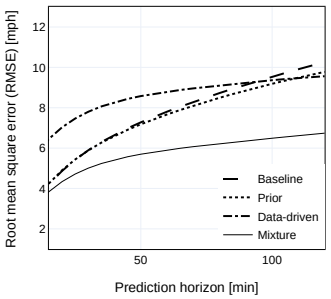
All datasets were normalized using the mean and standard deviation of each sensor in the training set. For a reference, we defined a baseline method that predicts future traffic features assuming that the current traffic does not change over time, i.e., $\mathbf{x}_{x+h|t}^{\text{baseline}} = \mathbf{x}_t$.

3.4.2 Analysis of network prior

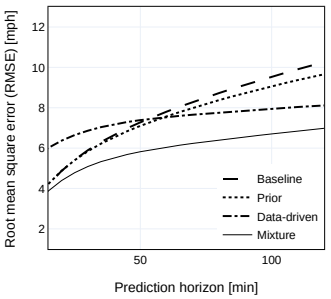
In this section, we show how network prior information contributes to predictive performance. Our model generalizes the DLM [56] to extend the model for a more extensive sensor network using the sensor's topology structure. When the DLM is simply used in an extensive network



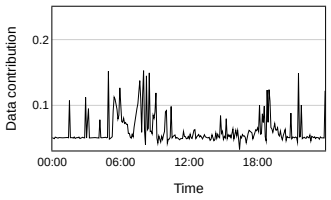
(a) Prediction accuracy (97/73 days for training/test set; 43:37 ratio)



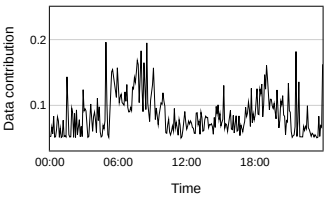
(b) Prediction accuracy (194/73 days for training/test set; 73:27 ratio)



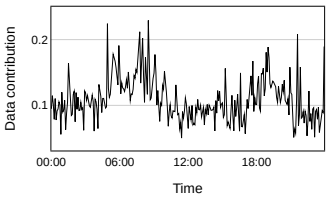
(c) Prediction accuracy (292/73 days for training/test set; 8:2 ratio)



(d) c_t^{data} (97/73 days for training/test set; 43:37 ratio)



(e) c_t^{data} (194/73 days for training/test set; 73:27 ratio)



(f) c_t^{data} (292/73 days for training/test set; 8:2 ratio)

Figure 3.5: Accuracy of the prediction and the data contribution for different training-test set ratio. The baseline method predicts future traffic features assuming that the current traffic does not change over time, i.e., $\mathbf{x}_{x+h|t}^{\text{baseline}} = \mathbf{x}_t$.

without topology structure information, an overfitting problem can occur. We introduce the three following setups to evaluate how well the proposed model utilizes the topology structure avoiding the overfitting problem,

1. a single DLM for the entire sensor network (without topological information),
2. separate DLMs ($K = 5$) for each freeway (block-diagonal DLM),
3. and the proposed model that is a single DLM ($K = 5$) with topological information.

As shown in Fig. 3.3, the proposed model shows the best performance, followed by block-diagonal DLM and single DLM without topological information. The proposed model induces the sensor's topological information through heat diffusion kernels to give weights to each element of this transition matrix and focus on estimating more essential components, resulting in it as a sparse matrix, as shown in Fig. 3.4. As a result, it shows excellent performance in long-term prediction by effectively estimating off-diagonal elements (correlation between signals of sensors installed on different freeways) while avoiding the overfitting problem. In the case of the model with a single DLM, all elements of this matrix are estimated using historical data, while in the case of the model with separate DLMs, only the block diagonal elements are estimated (red shaded area). Therefore, since the former one needs to estimate a

much larger number of elements from the data than the latter, an overfitting problem may occur. In contrast, in the separate DLMs, the historical data cannot be fully utilized due to the lack of association between sensors belonging to different freeways. In particular, this insufficiency causes degradation of long-term predictions as congestion propagates slowly from one freeway to others.

The low prediction error is obtained only when the topological information is optimally implanted into the DLM. Bayesian inference in our model is the key component to support this process, as it optimally estimates various parameters that characterize the mixing ratio between data and prior, which respectively correspond to DLM and topological information. We set up the following experiment to find test the effectiveness of this estimation method:

1. the model with measurements (Eq. (3.2)),
2. the model with topological information (Eq. (3.15)),
3. and the model with both topological information and measurements (Eq. (3.19)).

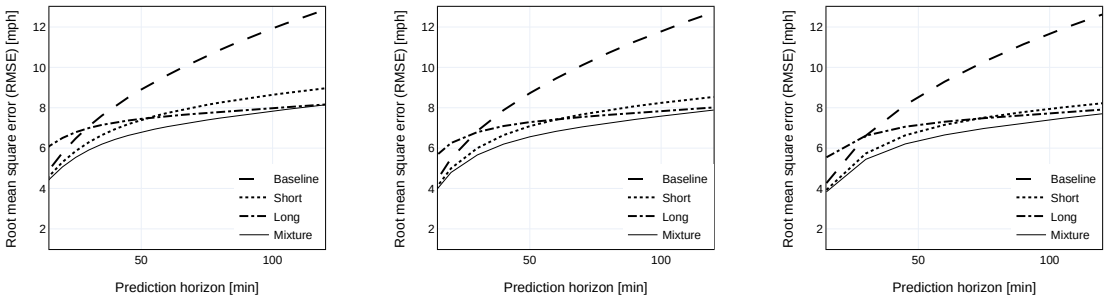
For all the above models, we set three different cases that are characterized by different sizes of the training sets with the same test set.

Figures 3.5(a)-(c) show the prediction accuracy of each case. Interestingly, the model with measurements produced smaller errors when the size of the training set is smaller. The reason is that each training set period is close to that of the test set with respect to time, which means larger training sets contain measurement that are far from those in the test sets. This may distort the inference process as traffic measurements have seasonal patterns. On the other hand, the model using only the topological information showed poor performance in predicting the far future because mixture kernels do not represent well the change in traffic conditions due to the volume preservation characteristic. The model with both topological information and measurements showed the best performance and similar outputs regardless of the size of the training set. It shows that Bayesian inference estimates parameters α_t and γ_t in Eq. (3.19) optimally, extracting maximal information both from data and prior.

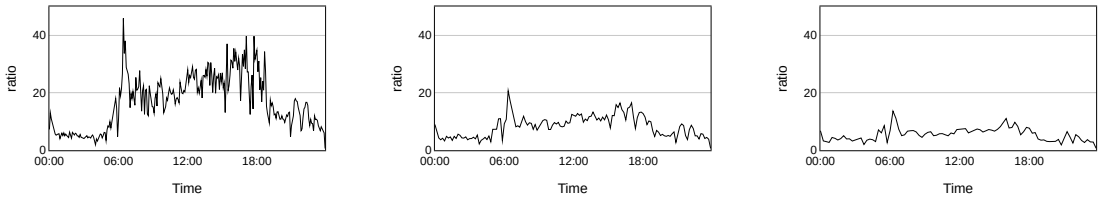
Figures 3.5(d)-(f) show the data contribution which is defined in Eq. (3.20) of the mixture model. As the size of the training set increases, the data contribution increases since the larger training set can generalize measurements more easily. Another important aspect from the results is that the data contribution increases during peak periods such as morning and evening peaks since the traffic volume is most likely not preserved during these periods (therefore, it is difficult to explain it only with diffusion processes).

3.4.3 Analysis of different diffusion periods

We evaluated the proposed method with different diffusion processes (short, long, and mixture of both) in order to examine how the model of Eq. (3.9) performs in different settings. The



(a) Prediction accuracy (T_s =5 min.) (b) Prediction accuracy (T_s =10 min.) (c) Prediction accuracy (T_s =15 min.)



(d) $\pi_t^{(\tau_0)}/\pi_t^{(\tau_\infty)}$ (T_s =5 min) (e) $\pi_t^{(\tau_0)}/\pi_t^{(\tau_\infty)}$ (T_s =10 min.) (f) $\pi_t^{(\tau_0)}/\pi_t^{(\tau_\infty)}$ (T_s =15 min.)

Figure 3.6: Accuracy of the prediction and ratio of the short and long diffusion processes for the same test set with different time intervals. The baseline method predicts future traffic features assuming that the current traffic does not change over time, i.e., $\mathbf{x}_{x+h|t}^{\text{baseline}} = \mathbf{x}_t$.

transition matrix $\hat{\mathbf{H}}_t$ was set from Eq. (3.19) with three different diffusion priors:

1. $\mathbf{H}_t^{\mathcal{G}}(\mathcal{T}) = \mathbf{H}^{\mathcal{G}}(\tau_0) = \lim_{\tau\to 0} e^{-\tau\mathbf{L}(\mathcal{G})}$ (short diffusion kernel; identity mapping),
2. $\mathbf{H}_t^{\mathcal{G}}(\mathcal{T}) = \mathbf{H}^{\mathcal{G}}(\tau_\infty) = \lim_{\tau\to\infty} e^{-\tau\mathbf{L}(\mathcal{G})}$ (long diffusion kernel; averaging),
3. and $\mathbf{H}_t^{\mathcal{G}}(\mathcal{T}) = \pi_t^{(\tau_0)}\mathbf{H}^{\mathcal{G}}(\tau_0) + \pi_t^{(\tau_\infty)}\mathbf{H}^{\mathcal{G}}(\tau_\infty)$ (mixture of short and long diffusion kernels).

We also set three different cases that are characterized by different sampling intervals (T_s), 5, 10, and 15 minutes. The sampling interval indicates the time duration that corresponds to the one-time incremental (the difference between $t+1$ and t). The sampling interval is related to the diffusion period τ as a diffusion kernel expresses how traffic signals diffuse through a graph within a sampling interval.

Figures 3.6(a)-(c) show the prediction accuracy of each diffusion prior on the transportation network $\mathcal{G}_1$ with the three different sampling intervals. The predictor with the long diffusion process showed relatively poor performance compared to the baseline method for small prediction horizons, but it was improved when prediction horizons become larger. On the other hand, the one with the short diffusion process showed relatively good performance compared to the baseline method for all prediction horizons; however, it had insufficient performance for large prediction horizons compared to the one with the long diffusion process. The mixture model takes advantage of the two extreme cases, significantly improving the

performance for both small and large prediction horizons. Specifically, around 50 minutes prediction horizon in Fig. 3.6(a), the performance of the mixture model is noticeably better than the others, meaning that a mixture of poor predictors can produce a good performance.

We emphasize that the distribution of the diffusion processes (Π_t) was determined optimally by Bayesian inference. Figures 3.6(d)-(f) show the ratio of the coefficients $\pi_t^{(\tau_0)}$ and $\pi_t^{(\tau_\infty)}$ in the mixture model that corresponds to the short and long diffusion processes, respectively. Although the short diffusion process dominates the whole process, as shown in the figures, the small portion of the long diffusion process contributes to the improvement. More importantly, the ratio becomes smaller when the sampling interval increases. It shows that Bayesian inference performs well in optimally determining parameters, since the performance of the mixture model stays similar when the sampling interval is changed.

We also emphasize that the ratio depends on time. For example, during the early morning, the diffusion kernel with long diffusion period (τ_∞) contributes more to the prediction performance although short diffusion (identity mapping) seems to be a more reasonable choice as there are few changes in traffic during that time. However, if the signal values are relatively uniform (in the case of a traffic signal at early morning), taking an average can remove noise while minimizing signal distortion as $\mathbf{x}_{t+1} \approx \mathbf{x}_t$ (identity) $\approx \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{x}_t$ (averaging; robust to noise).

3.4.4 Comparison with state-of-the-art technologies

We compare the proposed method with other methods using a benchmark dataset: PEMS-BAY dataset [59]. For a fair comparison, we use the same settings which are defined in [59] (also same in [67])[2]. The models used for the comparison are as follows.

FC-LSTM (Fully Connected Long Short-Term Memory)

This model has been used as a representative reference for time-sequence modeling in deep learning [75]. In general, the LSTM module extracts correlations of signals farther apart in time than the RNN structure. However, this model's disadvantage is that spatial correlations can only be expected to learn directly from data as there is no separate module for extracting spatial relationships of signals. The RMSE score for PEMS-BAY dataset is retrieved from [59].

STGCN

Yu, Yin, and Zhu [63] extracted spatial features with Graph Convolutional Neural Network (CNN) utilizing spectral graph convolution in graph theory. After that, they attached Gated CNN block to extract temporal features.

[2] Our code is available at: https://github.com/semink/lsdlm/

Table 3.2: RMSE of different methodsfor PEMS-BAY dataset.

Horizon	15 min	30 min	60 min
FC-LSTM [75]	4.19	4.55	4.96
DCRNN [59]	2.95	3.97	4.74
STGCN [69]	2.96	4.27	5.69
Graph WaveNet [67]	**2.74**	**3.70**	4.52
ST-MetaNet [76]	2.90	4.02	5.06
Proposed	2.90	3.77	**4.44**

DCRNN (Diffusion Convolution Recurrent Neural Network)

Li, Yu, Shahabi, *et al.* [59] constructed a successful predictor by extracting the signal's spatial features from the underlying graph structure by diffusion convolutional layers. Compared to STGCN, they designed the filter in the spatial domain directly rather than the graph spectral domain. The authors combine this diffusion module to Gated Recurrent Unit (GRU) which is a Recurrent Neural Network (RNN) variant.

Graph WaveNet

Xu, Shen, Cao, *et al.* [67] improved DCRNN by using dilated 1D convolution (also called WaveNet) to extract temporal features in terms of computation time and performance.

ST-MetaNet

Pan, Liang, Wang, *et al.* [76] introduced graph attention network to extract spatial features. They utilize RNN architecture to extract temporal features.

Table 3.2 shows the RMSE of each model and our proposed method. We confirm that the performance of the proposed method reaches that of state-of-the-art methods based on a complex deep learning architecture. It even performs better for long-term prediction as we model based on DLM that explicitly expresses the daily periodicity of traffic signals. For example, the RMSEs of our proposed method for 90 and 120 min horizons are respectively 4.70 and 5.26, while these are 5.26 and 6.02 with the pre-trained DCRNN model.[3]

Our proposed method requires lower computational effort compared to the others. Also, it infers the majority of the parameters (N^2) analytically by Eq. (3.19). The method only requires numerical computation when it solves the optimization problem (3.25) to infer $K+2$ parameters, which has $O(K^2)$ complexity, where K^2 is noticeably smaller than N^2. Note that the hyperparameters are optimally estimated by solving the optimization problem (3.25) rather than the cross-validation method. As hyperparameter tuning is an expensive task, it

[3] As GraphWaveNet predicts all the horizons at once (not recursive), we could not use the pre-trained model for the longer horizons. As a result, we choose DCRNN which shows the second-best result on 60 min horizon.

Table 3.3: Computation costs for training on the PEMS-BAY dataset

Model	Training(s)
DCRNN [59]	750 (per epoch)
Graph WaveNet [67]	580 (per epoch)
Proposed	760 (total)

can be a major advantage of the proposed method.

On the other hand, all state-of-the-art methods require heavy numerical computations to train a large number of parameters as they are based on deep-neural-net architectures. Our method successfully infers all parameters at the time scale of minutes with CPU computations, which is noticeably shorter than other DNN based methods with GPU computations as shown in Table 3.3 (note that the DNN based methods required from 50 epochs to 100 epochs to converge).

Another advantage of our model compared to the deep-learning-based architectures is that only a small number of parameters need to be decided heuristically. This can provide easy scalability to apply our model to other traffic datasets or datasets with similar properties to traffic data (daily periodicity). For example, in our model, the parameters to be determined before training are the threshold constant κ, the kernel width σ to build a proper graph, and the number of diffusion processes K to determine how many diffusion processes should be mixed. We empirically choose the constants κ and σ such that the corresponding graph $\mathcal{G}$ is a k-vertex-connected graph with a small number k. For the number of diffusion processes K, we set $K = 5$ for the PEMS-BAY dataset but the prediction performance is not sensitive to the parameter (± 0.01 minutes changes of the RMSE score from $K = 3$ to $K = 7$).

3.5 Conclusion

In this chapter, we proposed a method for predicting traffic signals in transportation sensor networks. We successfully integrated topological information of the sensor network into a data-driven model by assuming that the parameters in the model are supported by the mixture of diffusion kernels with uncertainties. We exploited the Bayesian inference to optimally determine the parameters that characterize the distribution of diffusion processes and the importance of measurements against prior information. The importance varies with time, and we discover that the data are relatively more important, especially for the peak period. Most importantly, the proposed method reached accurate prediction at the level of state-of-the-art methods with less computational effort. It particularly shows excellent performance in long-term predictions by exploiting DLM's periodicity modeling. Our method can be applicable for predicting graph signals exhibiting daily patterns such as weather or energy consumption. For future works, we may improve the short-term prediction performance if we give more valuable prior information (e.g., graph structure more suitable for prediction; currently, it

only depends on topology), or if it is possible to derive all inference processes (especially the marginalization steps in Eq. (B.9) and (B.4)) with a non-linear model overcoming the limitation of linear models.

4 Traffic forecasting with a deep neural network model

This chapter is written with its own notations, and the content of this chapter is available as a submitted paper:

> Semin Kwak, Danya Li and Nikolas Geroliminis. "TwoResNet: Two-level resolution neural network for traffic forecasting of freeway networks". In: (Macau, China, Otc. 8-12, 2022). IEEE ITSC 2022 - 25th IEEE International Conference on Intelligent Transportation Systems, 2022 (submitted)

4.1 Introduction

Most of the works related to traffic forecasting in freeway networks have recently designed forecasters based on deep learning architectures [77]–[83]. The state-of-the-art performances in traffic prediction are also mostly acheived by deep learning-based methods, with only a few exceptions [84].

Spatial correlations of different sensors are crucial components for achieving an accurate prediction. For example, when trying to predict traffic features of a sensor, those of neighboring sensors give great information since congestion propagates through road networks [85]. Graph convolutional network (GCN) is one of the most popular architectures that captures such correlations; therefore, most state-of-the-art methods implant it into their architectures [79], [81]–[83], achieving accurate prediction results.

Diffusion convolutional recurrent neural network (DCRNN) [81] is one of the pioneering works that adopt GCN for traffic forecasting. The authors successfully combined a GCN architecture with a recurrent neural network (RNN) and proved that extracting spatial correlations is critical for improving prediction accuracy. Graph WaveNet [82] further improves the way to extract spatial correlations. The authors revealed that the GCN is based on predefined road connectivity, and the predefined one may miss the latent connectivity. Therefore, they

suggested a new architecture that can learn road connectivity directly from data in an end-to-end fashion. As a result, Graph WaveNet achieves remarkable performance improvements, especially for long-term predictions.

This significant breakthrough in Graph WaveNet suggests that a hidden key parameter for long-term predictions can be found by comparing the spatial correlations extracted by GCN with those learned from data. Conventional GCN is known to extract spatial correlations among neighboring nodes remarkably well, but not with remote nodes [11]. The underlying reason is a practical limitation in the size of receptive fields during training processes. Receptive field size has to be limited from a training perspective because it is related to the depth of a network; the deeper a network is, the harder it is to be trained. On the other hand, Graph WaveNet learns road connectivity from data, allowing GCN to extract correlations among remote sensors. Such a contrast provides an insight that properly defining connectivity among remote sensors is essential in long-term traffic predictions. However, this insight is neither explicitly described in the original paper nor tested in the literature.

Based on the intuition, we introduce a deep neural network architecture called the two-level resolution neural network (TwoResNet) that consists of two main blocks: the low-resolution and high-resolution blocks. Each block focuses on macroscopic and microscopic traffic dynamics, respectively. When the high-resolution block predicts microscopic traffic, it refers to a macroscopic prediction from the low-resolution block. In other words, it allows the GCN in the high-resolution block to refer to remote sensors at a coarsened level. We explain how TwoResNet works with an analogy of how a company works. The boss of a company (low-resolution block) focuses more on long-term visions while employees (high-resolution block) focus more on short-term objectives, keeping the boss' long-term ideas in their mind, to maximize both short and long-term profits. As a result, our TwoResNet achieves a competitive prediction accuracy compared to state-of-the-art methods.

The main contributions of this work are as follows:

- We suggest TwoResNet[1], which learns macro and microscopic traffic dynamics separately through the low-resolution and high-resolution blocks.

- We introduce a clustering method to group sensors for acquiring macroscopic traffics, considering both proximity and correlations simultaneously.

- TwoResNet achieves a competitive prediction accuracy compared to most reliable state-of-the-art works. For the long-term predictions, we obtained remarkable improvements compared to the Graph WaveNet [82] without learning road connectivity.

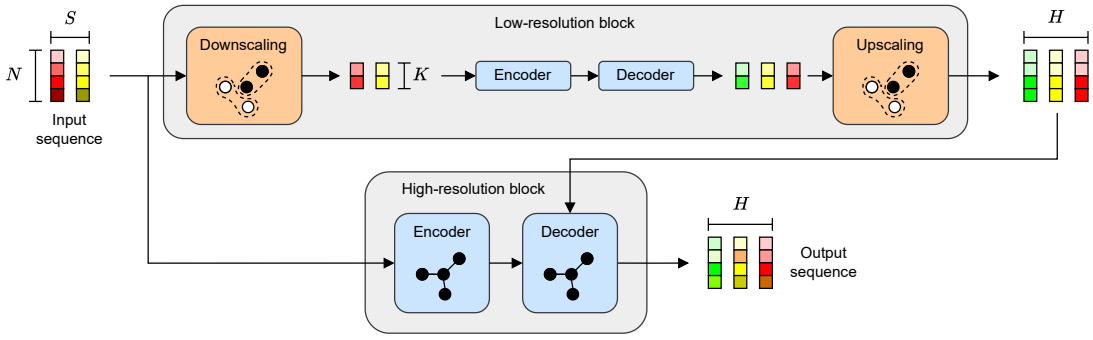


Figure 4.1: The architecture of a two-level resolution neural network (TwoResNet). In the figure, S, H, N, and K represent the input sequence length, the maximum prediction, the number of sensors, and the number of clusters, respectively.

4.2 Model architecture

We introduce our proposed model in a top-down approach. The two-level resolution network (TwoResNet) is a deep neural network model that predicts time series for multi-step horizons based on an input sequence, given predefined clusters and topological information:

$$y = \text{TwoResNet}(x|\text{cluster}, \text{topology}) \tag{4.1}$$

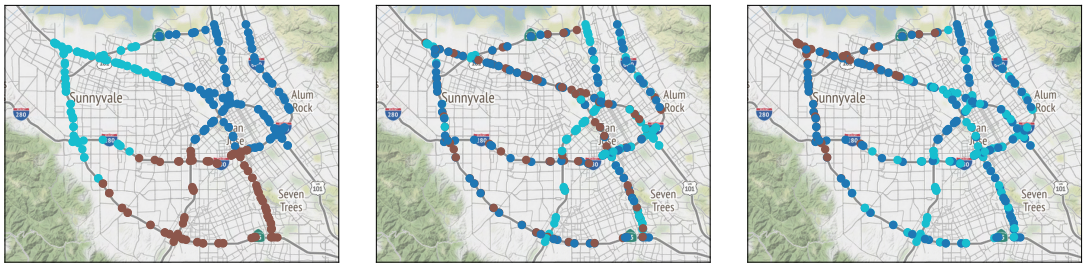
where the input sequence $x = (x_{-S+1}, \ldots, x_0) \in \mathbb{R}^{S \times F_{\text{in}} \times N}$ (S, F_{in}, and N are the input sequence length, the number of input features, and the number of sensors, respectively) and the output sequence $y = (y_1, \ldots, y_H) \in \mathbb{R}^{H \times F_{\text{out}} \times N}$ (H and F_{out} are the output sequence length, i.e., the maximum prediction horizon and the number of output features, respectively) that should be close enough to the ground truth $y^{\text{true}} = (y_1^{\text{true}}, \ldots, y_H^{\text{true}})$.

It consists of two core blocks: Given the clustering information of each sensor, a low-resolution block maps an input sequence x into an output sequence $\bar{y} = (\bar{y}_1, \ldots, \bar{y}_H) \in \mathbb{R}^{H \times F_{\text{out}} \times N}$ defined on a macroscopic scale. On the other hand, given topological information (i.e., road connectivity), the high-resolution block maps the input sequence x and the output sequence of the low-resolution block $\bar{y}$ into a desirable output sequence y:

$$\begin{aligned} \bar{y} &= \text{LowResolutionBlock}(x|\text{cluster}) \\ y &= \text{HighResolutionBlock}(x, \bar{y}|\text{topology}) \end{aligned} \tag{4.2}$$

Figure 4.1 shows the architecture and data flow of TwoResNet. Through this architecture, we induce the low-resolution and the high-resolution blocks to focus on capturing traffic dynamics on a macro and microscopic scale, respectively.

[1] https://github.com/semink/TwoResNet



(a) Proximity-based adjacency.

(b) Correlation-based adjacency.

(c) Mixture adjacency of proximity-based and correlation-based (α = 0.5).

Figure 4.2: Spectral clustering ($K = 3$) of sensors in a freeway network in with different adjacency matrices. Nodes with the same color belong to the same cluster.

4.2.1 Low-resolution block

We design the low-resolution block to predict traffic features on a macroscopic scale, such as an average speed by region. It consists of a Downscaling module, an encoder, a decoder, and an Upscaling module. Before an input sequence is fed into the encoder, the Downscaling module aggregates traffic features of N sensors into K dimension with predefined clusters (which will be explained later). After that, the encoder transforms the aggregated sequence into a higher representation. The decoder generates an output sequence based on the representation followed by the Upscaling module that broadcasts the K-dimensional features into the original dimension N:

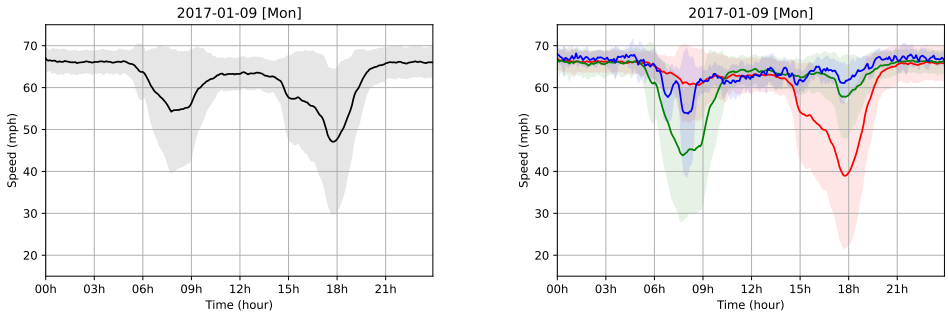
$$\begin{aligned} \dot{x} &= \text{Downscaling}(x|\text{cluster}) \\ \dot{z} &= \text{Encoder}(\dot{x}) \\ \dot{y} &= \text{Decoder}(\dot{z}) \\ \tilde{y} &= \text{Upscaling}(\dot{y}|\text{cluster}) \end{aligned} \tag{4.3}$$

Clustering

As seen in Eq. (4.3), the downscaling and upscaling modules require predefined clusters. For clustering, we utilize spectral clustering [86]. Spectral clustering is a simple but powerful clustering tool that very often outperforms classical clustering algorithms. Also, it has an outstanding property that clusters data defined on the non-euclidean domain. The algorithm takes an adjacency matrix of a graph structure as an input, together with the number of clusters K:

$$I = \text{SpectalClustering}(A, K) \tag{4.4}$$

where A is an adjacency matrix. The indicator matrix $I \in \mathbb{R}^{N \times K}$ such that $[I]_{n,k} = 1$ when the node n belongs to the cluster k, otherwise 0. Each node only belongs to one cluster exclusively.



(a) Without clustering. (b) With clustering.

Figure 4.3: The average speeds (a) of all sensors in the BAY area and (b) by clusters for the 9th of January 2017. In (b), each color represents one cluster. Shaded areas represent the standard deviation of each average speed of the same color.

For detailed explanation of the algorithm, see [87].

Figure 4.2(a) shows a clustering result ($K = 3$) with the Spectral clustering, based on an adjacency matrix A_{prox} that is defined as

$$[A_{\text{prox}}]_{i,j} = \begin{cases} d_{ij} & d_{ij} < \text{threshold} \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

where $d_{ij} = \exp\left(-\frac{\text{euclidean}^2(v_i,v_j)}{2\delta^2}\right)$, $\text{euclidean}(v_i, v_j)$ is the euclidean distance between the sensor i and j, and δ is a bandwidth coefficient. The threshold set a sparsity of the matrix. As shown in the figure, this clustering shows a strong locality; however, exploiting the averaged traffic features purely based on the proximity may not be ideal for prediction since sensors in the same group may have very different traffic patterns. For example, sensors on the same freeway in the opposite directions may show completely different patterns since people use one freeway during the morning but do the other during the evening for commuting.

With a correlation-based adjacency matrix A_{corr}, as shown in Fig. 4.2(b), sensors located in opposite directions are well separated. We defined the matrix A_{corr} similar to A_{prox} by defining $d_{ij} = \exp\left(-\frac{\text{correlation}(v_i,v_j)}{2\delta^2}\right)$, where $\text{correlation}(v_i, v_j)$ is the cross-correlation of the sensor i and j across the time. However, in this case, we lose the locality of each cluster since topological information is not involved in this process.

We mix proximity-based and correlation-based similarities to trade the locality and correlation appropriately. Therefore, we define another adjacency matrix A_{mix} such that:

$$d_{ij} = (1-\alpha)\exp\left(-\frac{\text{euclidean}^2(v_i,v_j)}{2\delta_1^2}\right) + \alpha\exp\left(-\frac{\text{correlation}(v_i,v_j)}{2\delta_2^2}\right) \tag{4.6}$$

with a hyperparameter α ($0 \le \alpha \le 1$).

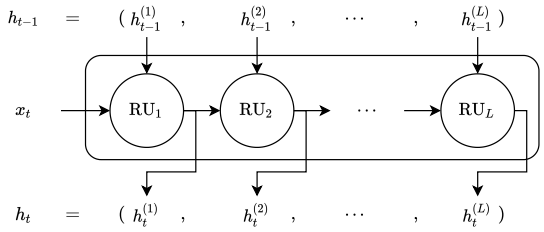


Figure 4.4: Stacked recurrent neural network.

Figure 4.2(c) shows a clustering result when $\alpha = 0.5$. Sensors in the opposite directions are grouped in different clusters, and each cluster also keeps topological locality at the same time. In order to evaluate the clustering, we obtain statistics on the average traffic flows before and after the clustering. Figures 4.3(a) and (b) show the average speeds of all sensors in the Bay area and by clusters for a day. Since the day is Monday, we can see clear morning and evening peaks (speed drop) in (a), and they become clearer after clustering in (b). For example, with $K = 3$, the sensors are grouped into whose measures (1) morning peak (green), (2) evening peak (red), (3) and free flow (blue).

The low-resolution block utilizes such clusters to aggregate features into a macroscopic level and processes them.

Downscaling and upscaling

Coarsening features is performed by averaging feature values that are within the same cluster. For an arbitrary $x_t \in \mathbb{R}^{F \times N}$ of a sequence, we define

$$\downarrow x_t = x_t \cdot I \cdot D_I^{-1} \in \mathbb{R}^{F \times K} \tag{4.7}$$

where D_I is a diagonal matrix having $e^T \cdot I$ as the diagonal term and e is the all one vector.

We also define an upscaling operation that simply broadcasts K dimensional macroscopic features into the original N dimension according to the clusters. For an arbitrary $\dot{x}_t \in \mathbb{R}^{F \times K}$,

$$\uparrow \dot{x}_t = \dot{x}_t \cdot I^T \in \mathbb{R}^{F \times N} \tag{4.8}$$

With the downscaling and upscaling operations, we design the modules

$$\begin{aligned} \text{Downscaling}(x|I) &= (\downarrow x_{-S+1}, \ldots, \downarrow x_0) \in \mathbb{R}^{S \times F_{\text{in}} \times K} \\ \text{Upscaling}(\dot{y}|I) &= (\uparrow \dot{y}_1, \ldots, \uparrow \dot{y}_H) \in \mathbb{R}^{H \times F_{\text{out}} \times N} \end{aligned} \tag{4.9}$$

Encoder and decoder

Both encoder and decoder are based on the stacked recurrent neural network (RNN) architecture [88]. The RNN architecture consists of L independent Recurrent Units (RU). The RNN recurrently process an input state (x_t) and stacked hidden states (h_{t-1}) and produce the next stacked hidden states (h_t):

$$h_t = \mathrm{RNN}(x_t, h_{t-1}) \tag{4.10}$$

Figure 4.4 shows the data flow when a stacked RNN processes an input sequence.

Therefore, the encoder of the low-resolution block consumes an input sequence one at a time, producing hidden states. Then the encoder repeats this process with the following input sequence until it processes all the series from $t = -S+1$ to $t = 0$:

$$\dot{h}_t = \mathrm{RNN}^{\mathrm{encoder}}_{\mathrm{low}}(\downarrow x_t, \dot{h}_{t-1}) \tag{4.11}$$

where $\dot{h}_{-S}$ is set to zero vectors for all L_{low} units. The last stacked hidden states $\dot{h}_0$ is set to the output of the encoder $\dot{z}$.

Similarly, the decoder generates stacked hidden states one at a time, recurrently from $t = 1$ to $t = H$, taking the previous output sequence and the decoder output of the last step:

$$\begin{aligned} \dot{h}_t &= \mathrm{RNN}^{\mathrm{decoder}}_{\mathrm{low}}(\dot{y}_{t-1}, \dot{h}_{t-1}) \\ \dot{y}_t &= \mathrm{Linear}(\dot{h}_t^{(L_{\mathrm{low}})}) \end{aligned} \tag{4.12}$$

where $\dot{y}_0 = \downarrow x_0$ and $\dot{h}_0 = \dot{z}$. The Linear module projects the hidden state of the last unit into the output dimension.

Recurrent unit

We use the L_{low} independent gated recurrent units (GRU) [89] for the recurrent units in the RNN architecture. For l-th GRU, we evaluate

$$\dot{h}_t^{(l)} = \mathrm{GRU}_l(\dot{h}_t^{(l-1)}, \dot{h}_{t-1}^{(l)}) \tag{4.13}$$

where the input state $\dot{h}_t^{(l-1)} \in \mathbb{R}^{F_i \times K}$, the hidden state $\dot{h}_{t-1}^{(l)} \in \mathbb{R}^{F_o \times K}$, and the next hidden state $\dot{h}_t^{(l)} \in \mathbb{R}^{F_o \times K}$. The internal processes of the GRU are as follows:

$$\begin{aligned} r_t^{(l)} &= \sigma(\mathbf{W}_{ir}^{(l)} \dot{h}_t^{(l-1)} + \mathbf{W}_{hr}^{(l)} \dot{h}_{t-1}^{(l)} + \mathbf{b}_r^{(l)} \cdot e^T) \\ z_t^{(l)} &= \sigma(\mathbf{W}_{iz}^{(l)} \dot{h}_t^{(l-1)} + \mathbf{W}_{hz}^{(l)} \dot{h}_{t-1}^{(l)} + \mathbf{b}_z^{(l)} \cdot e^T) \\ n_t^{(l)} &= \tanh(\mathbf{W}_{in}^{(l)} \dot{h}_t^{(l-1)} + \mathbf{b}_{in}^{(l)} \cdot e^T + r_t \odot (\mathbf{W}_{hn}^{(l)} \dot{h}_{t-1}^{(l)} + \mathbf{b}_{hn}^{(l)} \cdot e^T)) \\ \dot{h}_t^{(l)} &= (1 - z_t^{(l)}) \odot n_t^{(l)} + z_t^{(l)} \odot \dot{h}_{t-1}^{(l)} \end{aligned} \tag{4.14}$$

The bold-faced parameters are learnable parameters ($\mathbf{W}^{(l)} \in \mathbb{R}^{F_o \times F_i}$, $\mathbf{b}^{(l)} \in \mathbb{R}^{F_o}$), σ is the Sigmoid function, tanh is the hyperbolic tangent function, and $\odot$ is the element-wise product. For detailed explanation of each step in GRU, refer to [89]. We set $F_i = F_{\text{in}}$ and $F_o = F_{\text{hid}}$ for the first recurrent unit (GRU_1), and for the rest ($\text{GRU}_2 \sim \text{GRU}_{L_{\text{low}}}$) we set $F_i = F_{\text{hid}}$ and $F_o = F_{\text{hid}}$, where the number of hidden features F_{hid} is a hyperparameter. Note that GRU_l for the encoder and decoder does not share the parameters.

4.2.2 High-resolution block

The high-resolution block consists of an encoder and a decoder block similar to the low-resolution block. The entire architecture is motivated by [81]. The critical difference is in the decoder module that takes the output of the low-resolution block as an input, inducing the high-resolution block to focus on microscopic dynamics.

$$\begin{aligned} z &= \text{Encoder}(x|\text{topology}) \\ y &= \text{Decoder}(z, \bar{y}|\text{topology}) \end{aligned} \tag{4.15}$$

Encoder and decoder

Most encoder parts are identical to the low-resolution block, except it consists of different recurrent units that consider topological information. It processes all the input sequence from $t = -S+1$ to $t = 0$:

$$h_t = \text{RNN}_{\text{high}}^{\text{encoder}}(x_t, h_{t-1}) \tag{4.16}$$

where h_{-S} is set to all zero vector for all L_{high} units. The last stacked hidden state h_0 is set to the output of the encoder z.

The decoder of the high-resolution block has a similar structure to the encoder, except it accepts the output of the low-resolution block. We induce the decoder to capture traffic dynamics at the microscopic level by adding the difference between two consecutive predictions from the low-resolution block into the decoder input. Decoder iterates from $t = 1$ to $t = H$ generating y:

$$\begin{aligned} h_t &= \text{RNN}_{\text{high}}^{\text{decoder}}(y_{t-1} + \bar{y}_t - \bar{y}_{t-1}, h_{t-1}) \\ y_t &= \text{Linear}(h_t^{(L_{\text{high}})}) \end{aligned} \tag{4.17}$$

where $y_0 = x_0$, $\bar{y}_0 = \uparrow (\downarrow x_0)$, and $h_0 = z$. The Linear module maps the hidden features into the output dimension.

Adding $\bar{y}_t - \bar{y}_{t-1}$ into the decoder input induces the decoder to focus on microscopic components. Decomposing $y_t = \hat{y}_t + \tilde{y}_t$, where $\hat{y}_t$ and $\tilde{y}_t$ are the latent macro and microscopic components of y_t, and assuming the low-resolution block accurately predicts the macroscopic

component, i.e., $\hat{y}_t \approx \bar{y}_t$,

$$\begin{aligned} y_{t-1} + \bar{y}_t - \bar{y}_{t-1} &= \hat{y}_{t-1} + \tilde{y}_{t-1} + \bar{y}_t - \bar{y}_{t-1} \\ &\approx \hat{y}_t + \tilde{y}_{t-1} \end{aligned} \tag{4.18}$$

Therefore, we can interpret the decoder transforms the input $y_{t-1} + \bar{y}_t - \bar{y}_{t-1} \approx \hat{y}_t + \tilde{y}_{t-1}$ into the output $y_t = \hat{y}_t + \tilde{y}_t$. As a result, the decoder only learns the transformation of $\tilde{y}_{t-1}$ into $\tilde{y}_t$, which is the dynamics of microscopic components.

Recurrent unit

We use L_{high} graph convolutional gated recurrent unit (GCGRU) for the recurrent units in the RNN architectures both in the encoder and decoder. The l-th GCGRU evaluate

$$h_t^{(l)} = \text{GCGRU}_l(h_t^{(l-1)}, h_{t-1}^{(l)} | \text{topology}) \tag{4.19}$$

where the internal processes are

$$\begin{aligned} \check{h}_t^{(l-1)} &= \text{GCN}(h_t^{(l-1)} | \text{topology}) \\ \check{h}_{t-1}^{(l)} &= \text{GCN}(h_{t-1}^{(l)} | \text{topology}) \\ h_t^{(l)} &= \text{GRU}_l(\check{h}_t^{(l-1)}, \check{h}_{t-1}^{(l)}) \end{aligned} \tag{4.20}$$

Therefore, it first transforms an input state ($h_t^{(l-1)} \in \mathbb{R}^{F_i \times N}$) and a hidden state ($h_{t-1}^{(l)} \in \mathbb{R}^{F_o \times N}$) into higher representations ($\check{h}_t^{(l-1)} \in \mathbb{R}^{(2 \cdot M \cdot F_i + 1) \times N}$ and $\check{h}_{t-1}^{(l)} \in \mathbb{R}^{(2 \cdot M \cdot F_o + 1) \times N}$) that absorb topological information through the GCN (the following section will introduce the transformation in detail). Then, same as the low-resolution block, the transformed input and hidden state are fed into the GRU to extract temporal features, generating the next hidden state ($h_t^{(l)} \in \mathbb{R}^{F_o \times N}$). The parameters of GRU_l are not shared between the encoder and decoder.

Identical to the low-resolution block, we set $F_i = F_{\text{in}}$ and $F_o = F_{\text{hid}}$ for the first recurrent unit (GCGRU_1), and for the rest ($\text{GCGRU}_2 \sim \text{GCGRU}_{L_{\text{low}}}$), $F_i = F_{\text{hid}}$ and $F_o = F_{\text{hid}}$.

Graph convolutional network (GCN)

We use the spatial graph convolutional network (GCN) [81] for the GCN module. Given an adjacency matrix A with self-connection (normalized by each row) that is a representation of the topology of a freeway network, we transform features referring to neighboring node features by using the M-layered bidirectional graph convolution:

$$\begin{aligned} \text{GCN}(x|A, M) = \text{Concat}(x, \\ &\text{gconv}_1(x|A), \ldots, \text{gconv}_M(x|A), \\ &\text{gconv}_1(x|A^T), \ldots, \text{gconv}_M(x|A^T)) \end{aligned} \tag{4.21}$$

where $\text{gconv}_m(x|A)$ is a graph convolutional operation and Concat is the tensor concatenation in the feature dimension.

We define the graph convolutional operation adopting [81] as follows:

$$\text{gconv}_m(x|A) = x \cdot (T_m(A))^T \tag{4.22}$$

where $T_m(A)$ is the Chebyshev polynomials of the first kind that are obtained from the recurrence relation:

$$\begin{aligned} T_0(A) &= I \\ T_1(A) &= A \\ T_n(A) &= 2A \cdot T_{n-1}(A) - T_{n-2}(A) \quad (n \geq 2) \end{aligned} \tag{4.23}$$

With the definition, the input x is transformed by referring to the sensors within exactly m-hop neighbors.

4.3 Training

4.3.1 Loss

We set the mean absolute error (MAE) overall horizons as our training metric. During the training, we add the loss of the low-resolution block to the total loss of the network by a weight γ. This induces the high-resolution block to only focus on the microscopic dynamics since it makes the assumption in Eq. (4.18) valid:

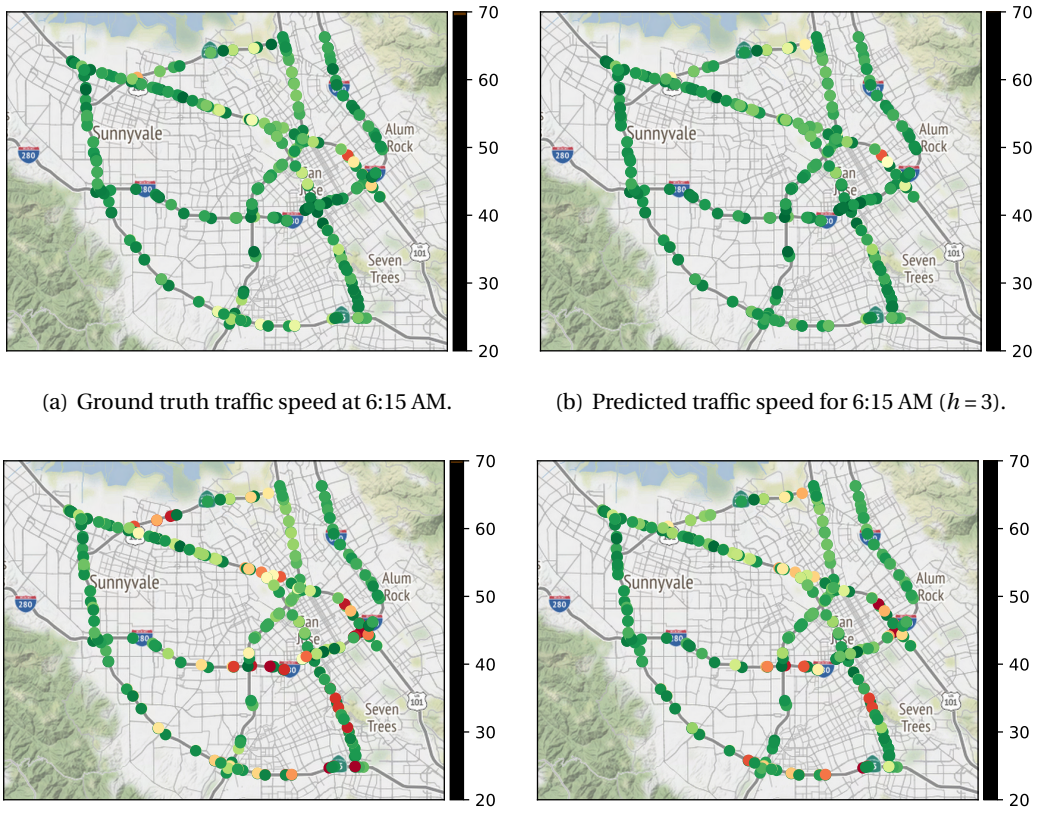
$$\begin{aligned} \text{MAE}_{\text{low}} &= \text{MAE}(\bar{y}, \uparrow (\downarrow y^{\text{true}})) \\ \text{MAE}_{\text{high}} &= \text{MAE}(y, y^{\text{true}}) \\ \text{training loss} &= \text{MAE}_{\text{high}} + \gamma \cdot \text{MAE}_{\text{low}} \end{aligned} \tag{4.24}$$

where

$$\text{MAE}(a, a^{\text{true}}) = \frac{1}{H \cdot F_{\text{out}} \cdot N} \sum_{h,f,n} |[a]_{h,f,n} - [a^{\text{true}}]_{h,f,n}| \tag{4.25}$$

4.3.2 Teacher forcing

We exploit a teacher forcing algorithm for training. Teacher forcing is an algorithm to feed observed sequence (i.e., ground-truth) back into the RNN after each step, forcing the RNN to stay close to the ground truth sequence. Therefore, we feed the ground truth y_{t-1}^{true} instead of the prediction of the previous step y_{t-1} in Eq. (4.17) with a probability p. A desirable design for the probability is that it should be close to 1 at the early training phase and approach 0 after enough epochs [90]. We model it as an inverse Sigmoid, defining the half-life coefficient



(a) Ground truth traffic speed at 6:15 AM.

(b) Predicted traffic speed for 6:15 AM ($h = 3$).

(c) Ground truth traffic speed at 7:00 AM.

(d) Predicted traffic speed for 7:00 AM ($h = 12$).

Figure 4.5: Traffic speeds on 20th of June, 2017 and predicted speeds by TwoResNet (PEMS-BAY dataset). The time when prediction is performed is t =6:00 AM. The unit of speed is mph.

η and reduce rate ρ, which should be greater than 0:

$$p(i|\eta,\rho) = \frac{1}{1+e^{-4\rho(\eta-i)}} \tag{4.26}$$

where i is the epoch index. The probability is 1/2 when $i = \eta$, and the derivative of p w.r.t. i at $i = \eta$ is $-\rho$ ($\frac{dp}{di}|_{i=\eta} = -\rho$).

4.4 Experiments

We evaluate TwoResNet on two public datasets, METR-LA and PEMS-BAY [81]. METR-LA dataset contains four months of aggregated traffic speed data every 5 minutes on 207 sensors on the freeways of Los Angeles County. PEMS-BAY dataset contains six months of aggregated traffic speed data every 5 minutes on 325 sensors in the Bay area.

4.4.1 Baselines

For the baselines, we choose followings:

- ARIMA: Autoregressive integrated moving average model with Kalman filter [81] as a representative model-based predictor.
- FC-LSTM: RNN with fully connected long short-term memory units [81] as a representative deep neural network-based predictor that only extracts temporal correlations of sensors.
- DCRNN: Diffusion convolution recurrent neural network [81], which is a pioneering model that successfully extracts spatio-temporal correlation with deep neural network architecture.
- Graph WaveNet: Convolutional neural network (CNN)-based deep neural network [82] that overcomes the drawback of DCRNN by learning road connectivity in an end-to-end fashion.

4.4.2 Setups

We adopt the same data pre-processing procedures as in [81]. The adjacency matrix of each freeway is constructed based on the travel distance between two sensors with the threshold Gaussian kernel [4]. The dataset is split in chronological order with 70% for training, 10% for validation, and 20% for testing. The input sequence length S and the maximum prediction horizon H both are set to 12 (1 hour input windows and 1 hour maximum prediction horizon). Z-score normalization is applied to the speed data.

We add time of the day and day of the week information into the input sequence since traffic features are highly periodic. Specifically, we encode time of the day into a vector $(\cos\xi, \sin\xi)$, where $\xi = 2\pi(1/24 * \text{hour} + 1/60 * \text{minute})$. With the polar encoding, the distance between any vector pairs of time ξ_1 and ξ_2 is constant if $|\xi_1 - \xi_2|$ is fixed. For day of the week information, we map weekdays and weekends into 0 and 1, respectively.

For the low-resolution block, we group all the sensors into 5 clusters based on Eq. (4.4). For the number of clusters, we choose it by the eigengap heuristic [87]. Cross-correlation is used to calculate the correlation-based adjacency matrix with the training set. The threshold was set such that the 90% of the adjacency matrix have zero values. The bandwidth parameters δ_1 and δ_2 are set to the standard deviations of corresponding distances $(\text{euclidean}^2(v_i, v_j), \text{correlation}(v_i, v_j))$. For both blocks, we set $F_{\text{hid}} = 64$ for the recurrent units. The GCN parameter M is set to 2. We also set the teacher forcing only for the high-resolution block and the weight γ for the MAE_{low} is set to 1.

All the other detailed settings for the hyperparameters are found in https://github.com/semink/TwoResNet.

Table 4.1: Performance comparison of TwoResNet and baseline methods.

Data	Models	15 min			30 min			60 min		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
METR-LA	ARIMA	3.99	8.21	9.60%	5.15	10.45	12.70%	6.90	13.23	17.40%
	FC-LSTM	3.44	6.30	9.60%	3.77	7.23	10.90%	4.37	8.69	13.20%
	DCRNN	2.77	5.38	7.30%	3.15	6.45	8.80%	3.60	7.60	10.50%
	Graph WaveNet	2.69	5.15	6.90%	3.07	6.22	8.37%	3.53	7.37	10.01%
	TwoResNet (ours)	**2.67**	**5.13**	**6.83%**	**3.05**	**6.18**	**8.13%**	**3.49**	**7.32**	**9.67%**
PEMS-BAY	ARIMA	1.62	3.30	3.50%	2.33	4.76	5.40%	3.38	6.50	8.30%
	FC-LSTM	2.05	4.19	4.80%	2.20	4.55	5.20%	2.37	4.96	5.70%
	DCRNN	1.38	2.95	2.90%	1.74	3.97	3.90%	2.07	4.74	4.90%
	Graph WaveNet	1.30	2.74	2.73%	1.63	3.70	3.67%	1.95	4.52	4.63%
	TwoResNet (ours)	**1.30**	**2.73**	**2.71%**	**1.62**	**3.70**	**3.60%**	**1.91**	**4.47**	**4.49%**

4.4.3 Results

Evaluation metrics

Following the convention, three different metrics are used to evaluate the performance: MAE (Eq. (4.25)), root mean square error (RMSE) and mean absolute percentage error (MAPE):

$$\begin{aligned} \mathrm{RMSE}(a, a^{\mathrm{true}}) &= \frac{1}{H \cdot F_{\mathrm{out}} \cdot N} \sum_{h,f,n} \left([a]_{h,f,n} - [a^{\mathrm{true}}]_{h,f,n} \right)^2 \\ \mathrm{MAPE}(a, a^{\mathrm{true}}) &= \frac{1}{H \cdot F_{\mathrm{out}} \cdot N} \sum_{h,f,n} \left| \frac{[a]_{h,f,n} - [a^{\mathrm{true}}]_{h,f,n}}{[a^{\mathrm{true}}]_{h,f,n}} \right| \end{aligned} \tag{4.27}$$

Comparison with baselines

We test all the baselines and our TwoResNet on both datasets for 15 minutes, 30 minutes, and 60 minutes ahead prediction. Table 4.1 summarizes the prediction accuracy for all the models. In general, TwoResNet shows the best scores among all horizons for all three metrics on the two datasets.

Figure 4.5 shows a sample prediction result of the TwoResNet for a peak period of $h = 3$ (15 minutes after the current time) and $h = 12$ (one hour after the current time). We can see the model predicts the beginning of congestion even though the input does not contain enough sign for the congestion.

Comparing ARIMA and FC-LSTM, we confirm that even a vanilla type of neural network is superior to the model-based predictor in terms of prediction accuracy. We also ensure how important to extract spatial correlation by comparing the results of FC-LSTM and DCRNN/-Graph WaveNet. On the other hand, the outstanding performance of our model proves that the

Table 4.2: Performance of HighResNet and TwoResNet on different time periods of the PEMS-BAY dataset.

Models	Period	15 min			30 min			60 min		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
HighResNet	Morning peak	1.85	3.69	4.35%	2.49	5.30	6.27%	3.08	6.57	8.12%
	Evening peak	1.74	3.54	4.16%	2.30	4.98	5.89%	2.73	5.99	7.26%
	Off-peak	1.00	2.17	1.74%	1.15	2.69	2.16%	1.34	3.21	2.65%
TwoResNet	Morning peak	**1.81**	**3.61**	**4.21%**	**2.41**	**5.09**	**5.94%**	**2.98**	**6.29**	**7.70%**
	Evening peak	**1.72**	**3.48**	**4.09%**	**2.24**	**4.78**	**5.64%**	**2.61**	**5.68**	**6.90%**
	Off-peak	**0.99**	**2.16**	**1.73%**	**1.14**	**2.61**	**2.08%**	**1.31**	**3.07**	**2.53%**

two-level resolution architecture successfully learns additional representations that could not be extracted through conventional GCN. Superb MAPE results prove that TwoResNet produces a more reliable prediction when congestion happens since the absolute percentage error is significantly higher under the condition than in the free flow situation with the same error (the same numerator).

Importance of the low-resolution block

We divide the prediction period into the morning, evening, and off-peak periods to analyze the importance of the low-resolution block in TwoResNet in detail. Each period is defined as 6 am to 10 am, 2 pm to 8 pm, and the rest of the period, respectively. We test the HighResNet and TwoResNet on each period of the PEMS-BAY dataset, where the HighResNet is a TwoResNet only with the high-resolution block. Based on the result summarized in Table 4.2, we conclude that adding the low-resolution block significantly improves prediction accuracy, mainly for the peak periods and the long-term predictions. With the two-resolutions architecture, the high-resolution block can focus more on short-term prediction while the low-resolution block does more on the long-term. We can interpret it by an analogy that a company only with employees (HighResNet) may find it difficult to have long-term plans (long-term predictions) and venerable to external shocks (peak periods) compared to a company that has intelligent supervisors (the low-resolution block in TwoResNet).

Ablation study of the periodicity encoding

We also conduct an ablation study for the periodicity encoding (time of the day and day of the week): We set TwoResNet with and without the periodicity encoding and evaluated them on the PEMS-BAY dataset. Table 4.3 shows the prediction results of the two settings. With the periodicity encoding, the prediction accuracy of the TwoResNet is improved for all prediction horizons. It proves that the periodicity encoding captures intrinsic periodicity, which cannot be extracted directly from the data.

Table 4.3: Performance of TwoResNet with and without the periodicity encoding (PE).

Models	15 min			30 min			60 min		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Without PE	1.31	2.76	2.74%	1.64	3.74	3.66%	1.94	4.54	4.60%
With PE	**1.30**	**2.73**	**2.71%**	**1.62**	**3.70**	**3.60%**	**1.91**	**4.47**	**4.49%**

4.5 Conclusion

We proposed a two-level resolution neural network model (TwoResNet) that is decomposed into a low-resolution and high-resolution block. The low-resolution block extracts regional representative values based on a clustering method and produces predictions of the regional changes. On the other hand, the high-resolution block predicts traffic features at the microscopic level, being aware of the expected regional changes, i.e., the output of the low-resolution block. As a result, TwoResNet achieves competitive scores compared to state-of-the-art methods on standard benchmark datasets. In future work, we will generalize the two-level architecture to a multi-level one, expecting it to learn more dynamic spatio-temporal correlations.

5 Conclusion and future research

5.1 Summary of contributions

In Chapter 2, we presented a dynamic linear model representing traffic dynamics on a small-scale freeway network. We effectively modeled non-linear traffic dynamics by expressing this dynamic linear model with time-varying parameters. By exploiting mathematical properties of the linear model, (1) we introduced a forgetting factor in the estimation model of time-varying parameters, giving more weight to the latest data, (2) analytically derived the optimal solution of the estimators from the least-squares perspective, (3) and, as a result, reduced the estimation time dramatically by analytically updating the estimated parameters with new data. In addition, by setting the time-varying parameters of the model to have a daily cycle (irrespective of the day of the week), unlike heuristic methods using different models for each major day, our prediction model significantly improved the prediction performance with an algorithmic approach. When we predicted the estimated travel time of a fixed freeway section using the proposed model, it outperforms other baselines for short-term prediction regardless of traffic situations.

In Chapter 3, we described how to generalize the dynamic linear model defined in Chapter 2 to the case of extensive freeway networks. In the original dynamic linear model, the number of parameters to be estimated is proportional to the square of the number of sensors in the network. Therefore, as the size of the network increases, the number of measurements relative to the number of parameters is relatively small, resulting in overfitting. To prevent this, parameters were defined using network topology information regardless of data. We induced parameters to fit the gap between this a priori model and the actual latent model with measurements. Here, the a priori model defines the relationship between sensors as a graph and then expresses it as a weighted sum of the graph diffusion kernels with different diffusion coefficients. This diffusion coefficient was learned from the data with Bayesian inference. Although a numerical process was required to optimize hyperparameters, learning was significantly faster than other deep neural network-based methods because most of the learning processes were performed analytically, thanks to the linear nature of DLM. In

particular, unlike neural network-based methods that optimize hyperparameters by exhaustive search-based algorithms, we elegantly optimized them by exploiting the marginalization process of Bayesian inference. These optimal hyperparameters have physical meanings, and therefore they explain which traffic characteristics are more important than the others. In addition, we showed excellent performance in long-term prediction compared to well-known deep neural network methods.

In the last chapter, we proposed a neural network model to express the traffic dynamics in extensive freeway networks, assuming that it shows complex intertwined patterns in both time and space, which linear models might be limited to describe. In particular, we recognized that the dynamics at macro and microscopic levels could have different behaviors. Therefore, we introduced TwoResNet, which predicts traffic at the two different levels separately (but finally fused) as a low-resolution and a high-resolution block, respectively. Macroscopic traffic was defined as representative values for each region of a freeway network. A clustering method was proposed to obtain these values, considering the network topology and the time series of each sensor simultaneously. The low-resolution block processed these representative values based on recurrent neural network (RNN) architecture and predicted future ones. In contrast, the high-resolution block consumed the raw data with RNN and graph convolution, extracting spatiotemporal correlations and predicting traffic at a microscopic level. As a result, the TwoResNet achieved great long-term prediction by inserting the low-resolution output, that is, predicted traffic at a macroscopic level, into the high-resolution RNN decoder input, which indirectly expanded the receptive field of graph convolutional network (GCN) to the size of each cluster. In particular, the TwoResNet outperformed one of the state-of-the-art models, which implied that it overcomes the shortcomings of the existing GCN.

5.2 Future research

The existing graph convolutional network shows excellence in extracting the spatial correlations but has the limitation that it must have a small receptive field. These shortcomings affect the long-term performance of traffic prediction. Therefore, our future research directs toward expanding the receptive field more effectively.

A graph convolutional network transforms features defined on each sensor by referring to those of surrounding sensors based on a predefined topology. As a particular case, the graph convolutional network (with a self-connected adjacency matrix) transforms the features of each sensor, referring to those of all sensors within the k-hop neighbors, by performing the convolution operation k times. Therefore, theoretically, features of any arbitrary sensors can be referred to by performing the convolutional operation several times. However, it smooths the features with k degree, resulting in missing details. Therefore, ironically, the more we expand the receptive field to achieve better long-term prediction, the worse we get the overall prediction performance.

We have noticed that extracting features of nearby sensors in detail is critical for accurate

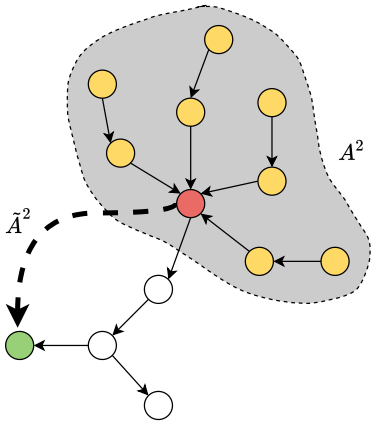


Figure 5.1: A new graph convolution.

short-term prediction, but extracting those of remote sensors on the same resolution is not necessary for an accurate long-term forecast. As a result, we defined the TwoResNet with two different resolutions. In addition to the two-resolution concept, we also believe that this structure can be extended to have a multilevel resolution one. By further subdividing the scope, we can classify freeway networks into three levels like a network/region/sensor or more. In this way, the information can flow from the macroscopic level to the microscopic level step by step (like TwoResNet), expecting the network learns more complex spatiotemporal correlation.

In the heart of this idea, we believe that clustering will play a decisive role in configuring the network in several resolutions. However, extending the clustering method used in TwoResNet may not be straightforward for a multi-resolution network. Although prediction accuracy was improved through clustering in TwoResNet, the clustering is not optimal for the actual objective, i.e., minimizing forecasting error. For example, it is not obvious for multi-resolution to decide what and how many potential representative values are in an intermediate resolution. Therefore, we believe clustering should be done in an end-to-end fashion. One expected challenge is that since all subsequent calculations will be performed based on this clustering result, if the network structure responsible for clustering is too complex, it would be challenging to allocate enough resources for prediction.

Another possible approach is to realize the multi-resolution concept without clustering by redefining the convolution operation of GCN. The new operation is expected to refer to (1) features of nearby neighbors in detail, (2) features of distant neighbors at a coarsened level, (3) and aggregated features of distant neighbors with minimal operation. The existing GCN mainly performs the convolution operation using a self-connected adjacency matrix (we call it A where all the diagonal term is 1). When we convolute features k times, i.e., $A^k x$, each feature is transformed by referring to the neighbors located within the k-hops. On the other hand, if the GCN performs the same operation using a self-disconnected adjacency matrix ($\tilde{A}$; all the diagonal term is 0), it only refers to the neighbors precisely k-hop distance away.

The first condition, (1), can be satisfied by using $\tilde{A}$. For example, if we parameterize $\tilde{A}x$ and $\tilde{A}^2x$, we can refer to neighbors that are 1 and 2-hop away in detail. On the other hand, if these two operations are combined, the conditions of (2) and (3) can be satisfied. If the convolution operation is performed m times using A and the same process is performed l times using $\tilde{A}$ in order, i.e., $\tilde{A}^l A^m x$, each node on the graph refers to features of l-hop neighbors whose features are already aggregated by m-hop neighbors. Figure 5.1 shows an example when $m = 2$ and $l = 3$. When the convolution operation is performed with A^m, the red-colored node aggregates all the features from the shaded area (m-hop neighbors). After that, the green-colored node refers the red-colored node by operating the convolution operation with $\tilde{A}^l$, resulting in it referring to the remote area in an aggregated way.

Although we expect to implement a more accurate predictor with this new operation, this method also challenges that the graph should be a directed graph: Most of the assumptions above will not be held if the graph is undirected. Therefore it may not be easy to generalize this operation for other tasks, although freeway networks are often modeled as a directed graph. Nevertheless, we expect that this challenge can be resolved since the new operation has great flexibility that we can mix the two matrices A and $\tilde{A}$ without any limitation.

A Appendix for Chapter 2

A.1 Mathematical derivations

A.1.1 Regularized least squares solution

A derivative of a scalar function is:

$$df(x,y) = \frac{\partial f}{\partial x}dx + \frac{\partial f}{\partial y}dy. \tag{A.1}$$

This can be extended to a matrix form as follows:

$$\begin{aligned} df(H) &= df\left(H_{1,1}, H_{2,1}, \cdots H_{n,m}\right) && \text{(A.2)}\\ &= \frac{\partial f}{\partial H_{1,1}}dH_{1,1} + \frac{\partial f}{\partial H_{2,1}}dH_{2,1} + \cdots \frac{\partial f}{\partial H_{n,m}}dH_{n,m} && \text{(A.3)}\\ &= vec^{\top}(dH)\cdot vec\left(\frac{df}{dH}\right) && \text{(A.4)}\\ &= \mathrm{tr}\left(dH^{\top}\frac{df}{dH}\right), && \text{(A.5)} \end{aligned}$$

where the function $vec(\cdot)$ vectorizes a matrix by concatenating its columns and

$$\frac{df}{dH} = \begin{bmatrix} \frac{\partial f}{\partial H_{1,1}} & \frac{\partial f}{\partial H_{1,2}} & \cdots & \frac{\partial f}{\partial H_{1,m}} \\ \frac{\partial f}{\partial H_{2,1}} & \frac{\partial f}{\partial H_{2,2}} & \cdots & \frac{\partial f}{\partial H_{2,m}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial H_{n,1}} & \frac{\partial f}{\partial H_{n,2}} & \cdots & \frac{\partial f}{\partial H_{n,m}} \end{bmatrix}, \tag{A.6}$$

$$dH = \begin{bmatrix} dH_{1,1} & dH_{1,2} & \cdots & dH_{1,m} \\ dH_{2,1} & dH_{2,2} & \cdots & dH_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ dH_{n,1} & dH_{n,2} & \cdots & dH_{n,m} \end{bmatrix}, \tag{A.7}$$

where $H_{i,j}$ denotes the i,j entry of matrix H.

We define the cost function of Eq. (2.8) as f:

$$f(H_k) \overset{\Delta}{=} \rho\lambda^{|\mathbb{D}|}\|H_k\|_F^2 + \left\|\left(V_{k+1}^{\mathbb{D}} - H_k V_k^{\mathbb{D}}\right)\Lambda_{|\mathbb{D}|}^{\frac{1}{2}}\right\|_F^2. \tag{A.8}$$

Since the cost function is convex [91], we utilize that the derivative at global minimum is zero. Therefore, we compute:

$$\begin{aligned} &f(H_k + dH_k) - f(H_k) \\ &= \mathrm{tr}\Big(2\Big(H_k V_k^{\mathbb{D}} \Lambda_{|\mathbb{D}|}\big(V_k^{\mathbb{D}}\big)^\top - V_{k+1}^{\mathbb{D}} \Lambda_{|\mathbb{D}|}\big(V_k^{\mathbb{D}}\big)^\top\Big) dH_k^\top \\ &\quad + H.O.T.\Big) + \rho\lambda^{|\mathbb{D}|}\mathrm{tr}\left(2H_k dH_k^\top + H.O.T.\right). \end{aligned} \tag{A.9}$$

Here, the abbreviation *H.O.T.* stands for higher order terms of dH_k. Assuming that dH_k is small enough,

$$\begin{aligned} &f(H_k + dH_k) - f(H_k) \\ &\quad = df(H_k) \\ &\quad = \mathrm{tr}\Big(2\Big(H_k V_k^{\mathbb{D}} \Lambda_{|\mathbb{D}|}\big(V_k^{\mathbb{D}}\big)^\top + \rho\lambda^{|\mathbb{D}|} H_k \\ &\qquad\qquad - V_{k+1}^{\mathbb{D}} \Lambda_{|\mathbb{D}|}\big(V_k^{\mathbb{D}}\big)^\top\Big) dH_k^\top\Big). \end{aligned} \tag{A.10}$$

The higher order terms are ignored since they are much smaller than the first order term dH_k. By Eq. (A.5), it is confirmed that:

$$\begin{aligned} \frac{df}{dH_k} = 2\Big(&H_k V_k^{\mathbb{D}} \Lambda_{|\mathbb{D}|}\big(V_k^{\mathbb{D}}\big)^\top \\ &+ \rho\lambda^{|\mathbb{D}|} H_k - V_{k+1}^{\mathbb{D}} \Lambda_{|\mathbb{D}|}\big(V_k^{\mathbb{D}}\big)^\top\Big). \end{aligned} \tag{A.11}$$

We set the derivative equal to zero to find the global minimum, then finally:

$$\tilde{H}_{k+1,k}^{\mathbb{D}} = V_{k+1}^{\mathbb{D}} \Lambda_{|\mathbb{D}|}\big(V_k^{\mathbb{D}}\big)^\top \Big(V_t^{\mathbb{D}} \Lambda_{|\mathbb{D}|}\big(V_k^{\mathbb{D}}\big)^\top + \rho\lambda^{|\mathbb{D}|} I_M\Big)^{-1}. \tag{A.12}$$

A.1.2 Regularization parameter

The data matrix $V_t^{\mathbb{D}} \Lambda_{|\mathbb{D}|} \left(V_k^{\mathbb{D}}\right)^\top$ in Eq. (2.10) can be decomposed as:

$$V_t^{\mathbb{D}} \Lambda_{|\mathbb{D}|} \left(V_k^{\mathbb{D}}\right)^\top = UDU^\top, \tag{A.13}$$

where $UU^\top = U^\top U = I_M$ and D is a diagonal matrix since the matrix is symmetric. Then, we can rewrite the inner part of the inversion in Eq. (2.10) as follows:

$$V_t^{\mathbb{D}} \Lambda_{|\mathbb{D}|} \left(V_k^{\mathbb{D}}\right)^\top + \rho\lambda^{|\mathbb{D}|} I_M = U\left(D + \rho\lambda^{|\mathbb{D}|} I_M\right) U^\top. \tag{A.14}$$

Equation (A.14) proves that even if the number of training data is not enough (i.e., there are some zero values in the diagonal of D), the inversion is still available since the regularization term $\left(\rho\lambda^{|\mathbb{D}|} I_M\right)$ is added and makes the regularized diagonal matrix $\left(D + \rho\lambda^{|\mathbb{D}|} I_M\right)$ all non-zero on the diagonal (= full rank). Therefore, the regularization term allows the model to be reliable in the inversion process.

A.1.3 Recursive update

We define two matrices $G_k^{\mathbb{D}}$ and $P_k^{\mathbb{D}}$ as follows:

$$G_k^{\mathbb{D}} \triangleq V_{k+1}^{\mathbb{D}} \Lambda_{|\mathbb{D}|} \left(V_k^{\mathbb{D}}\right)^\top, \tag{A.15}$$

$$P_k^{\mathbb{D}} \triangleq \left(V_k^{\mathbb{D}} \Lambda_{|\mathbb{D}|} \left(V_k^{\mathbb{D}}\right)^\top + \rho\lambda^{|\mathbb{D}|} I_M\right)^{-1}. \tag{A.16}$$

Then we rewrite Eq. (2.10) with the multiplication of these two matrices:

$$\tilde{H}_k^{\mathbb{D}} = G_k^{\mathbb{D}} P_k^{\mathbb{D}}. \tag{A.17}$$

The matrix $G_k^{\mathbb{D}\cup\tilde{d}}$ with a new day $\tilde{d}$, which does not belong to the training set $\mathbb{D}$, can be written as:

$$\begin{aligned}
G_k^{\mathbb{D}\cup\tilde{d}} &= V_{k+1}^{\mathbb{D}\cup\tilde{d}} \Lambda_{|\mathbb{D}\cup\tilde{d}|} \left(V_k^{\mathbb{D}\cup\tilde{d}}\right)^\top && \text{(A.18)}\\
&= \begin{bmatrix} V_{k+1}^{\mathbb{D}} & \mathbf{v}_{k+1}^{\tilde{d}} \end{bmatrix} \begin{bmatrix} \lambda\Lambda_{|\mathbb{D}|} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \left(V_k^{\mathbb{D}}\right)^\top \\ \left(\mathbf{v}_k^{\tilde{d}}\right)^\top \end{bmatrix} && \text{(A.19)}\\
&= \lambda V_{k+1}^{\mathbb{D}} \Lambda_{|\mathbb{D}|} \left(V_k^{\mathbb{D}}\right)^\top + \mathbf{v}_{k+1}^{\tilde{d}} \left(\mathbf{v}_k^{\tilde{d}}\right)^\top && \text{(A.20)}\\
&= \lambda G_k^{\mathbb{D}} + \mathbf{v}_{k+1}^{\tilde{d}} \left(\mathbf{v}_k^{\tilde{d}}\right)^\top && \text{(A.21)}
\end{aligned}$$

and

$$P_k^{\mathbb{D}\cup\tilde{d}} = \left(V_k^{\mathbb{D}\cup\tilde{d}}\Lambda_{|\mathbb{D}\cup\tilde{d}|}\left(V_k^{\mathbb{D}\cup\tilde{d}}\right)^\top + \rho\lambda^{|\mathbb{D}\cup\tilde{d}|}I_M\right)^{-1} \tag{A.22}$$

$$= \left(\begin{bmatrix} V_k^{\mathbb{D}} & \mathbf{v}_k^{\tilde{d}}\end{bmatrix}\begin{bmatrix}\lambda\Lambda_{|\mathbb{D}|} & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}\left(V_k^{\mathbb{D}}\right)^\top\\ \left(\mathbf{v}_k^{\tilde{d}}\right)^\top\end{bmatrix} + \rho\lambda^{|\mathbb{D}\cup\tilde{d}|}I_M\right)^{-1} \tag{A.23}$$

$$= \left(\lambda\left(V_k^{\mathbb{D}}\Lambda_{|\mathbb{D}|}\left(V_k^{\mathbb{D}}\right)^\top + \rho\lambda^{|\mathbb{D}|}I_M\right) + \mathbf{v}_k^{\tilde{d}}\left(\mathbf{v}_k^{\tilde{d}}\right)^\top\right)^{-1} \tag{A.24}$$

$$= \left(\lambda\left(P_k^{\mathbb{D}}\right)^{-1} + \mathbf{v}_k^{\tilde{d}}\left(\mathbf{v}_k^{\tilde{d}}\right)^\top\right)^{-1} \tag{A.25}$$

$$= \lambda^{-1}P_k^{\mathbb{D}} - \frac{\lambda^{-1}P_k^{\mathbb{D}}\mathbf{v}_k^{\tilde{d}}\left(\mathbf{v}_k^{\tilde{d}}\right)^\top P_k^{\mathbb{D}}\lambda^{-1}}{1+\lambda^{-1}\left(\mathbf{v}_k^{\tilde{d}}\right)^\top P_k^{\mathbb{D}}\mathbf{v}_k^{\tilde{d}}}, \tag{A.26}$$

where the derivations (A.25) to (A.26) are based on the matrix inversion lemma [92]. Eq. (A.21) and (A.26) show the availability of updating the matrices $\{P_k^{\mathbb{D}}, G_k^{\mathbb{D}}\}$ to $\left\{P_k^{\mathbb{D}\cup\tilde{d}}, G_k^{\mathbb{D}\cup\tilde{d}}\right\}$ with new measurements $\left\{\mathbf{v}_k^{\tilde{d}}, \mathbf{v}_{k+1}^{\tilde{d}}\right\}$. Therefore, we can estimate the new transition matrices $\bar{H}_k^{\mathbb{D}\cup\tilde{d}}$ with the updated matrices $\left\{P_k^{\mathbb{D}\cup\tilde{d}}, G_k^{\mathbb{D}\cup\tilde{d}}\right\}$ as in Eq. (A.17).

A.1.4 Linear minimum mean square estimator

A linear estimator for the velocity vector of i-step ahead at time t is written as:

$$\mathbf{v}_{k+i|k}^{\tilde{d}} = A_0\mathbf{v}_k^{\tilde{d}} + A_1\mathbf{v}_{k-1}^{\tilde{d}} + \cdots + A_{k-1}\mathbf{v}_1^{\tilde{d}} \tag{A.27}$$

What we need to do is to find an optimal set $\{A_0, A_1, \cdots, A_{k-1}\}$ in the minimum mean square error (MMSE) sense. From Eqs. (A.27) and Eq. (2.14), we define the prediction error as follows:

$$\begin{aligned}&\mathbf{v}_{k+i}^{\tilde{d}} - \mathbf{v}_{k+i|k}^{\tilde{d}}\\ &\quad= \left(\bar{H}_{k+i-1\leftarrow k}^{\mathbb{D}} - A_0\right)\mathbf{v}_k^{\tilde{d}} - \sum_{j=1}^{k-1}A_j\mathbf{v}_{k-j}^{\tilde{d}} + \mathbf{n}_{k+i-1\leftarrow k}^{\tilde{d}}\end{aligned} \tag{A.28}$$

Its mean square is

$$
\begin{aligned}
&\mathrm{E}\left[\left(\mathbf{v}_{k+i}^{\tilde{d}}-\mathbf{v}_{k+i|k}^{\tilde{d}}\right)\left(\mathbf{v}_{k+i}^{\tilde{d}}-\mathbf{v}_{k+i|k}^{\tilde{d}}\right)^{\top}\right] \\
&=\mathrm{E}\left[\left[\left(\bar{H}_{k+i-1\leftarrow k}^{\mathbb{D}}-A_0\right)\mathbf{v}_k^{\tilde{d}}-\sum_{j=1}^{k-1}A_j\mathbf{v}_{k-j}^{\tilde{d}}+\mathbf{n}_{k+i-1\leftarrow k}^{\tilde{d}}\right]\right. \\
&\qquad\left.\left[\left(\bar{H}_{k+i-1\leftarrow k}^{\mathbb{D}}-A_0\right)\mathbf{v}_k^{\tilde{d}}-\sum_{j=1}^{k-1}A_j\mathbf{v}_{k-j}^{\tilde{d}}+\mathbf{n}_{k+i-1\leftarrow k}^{\tilde{d}}\right]^{\top}\right] \\
&=\mathrm{E}\left[\left[\left(\bar{H}_{k+i-1\leftarrow k}^{\mathbb{D}}-A_0\right)\mathbf{v}_k^{\tilde{d}}-\sum_{j=1}^{k-1}A_j\mathbf{v}_{k-j}^{\tilde{d}}\right]\right. \\
&\qquad\left.\left[\left(\bar{H}_{k+i-1\leftarrow k}^{\mathbb{D}}-A_0\right)\mathbf{v}_k^{\tilde{d}}-\sum_{j=1}^{k-1}A_j\mathbf{v}_{k-j}^{\tilde{d}}\right]^{\top}\right] \\
&\qquad+\mathrm{E}\left[\mathbf{n}_{k+i-1\leftarrow k}^{\tilde{d}}\left(\mathbf{n}_{k+i-1\leftarrow k}^{\tilde{d}}\right)^{\top}\right]
\end{aligned}
\tag{A.29}
$$

and Eq. (A.29) is minimized when

$$
A_0=\bar{H}_{k+i-1\leftarrow k}^{\mathbb{D}} \tag{A.30}
$$

and

$$
A_j=0 \text{ for } j=1,2,\cdots,k-1. \tag{A.31}
$$

Therefore, the linear MMSE estimator for the velocity vector of i-step ahead at time step k is:

$$
\mathbf{v}_{k+i|k}^{\tilde{d}}=A_0\mathbf{v}_k^{\tilde{d}} \tag{A.32}
$$

$$
=\bar{H}_{k+i-1\leftarrow k}^{\mathbb{D}}\mathbf{v}_k^{\tilde{d}}, \tag{A.33}
$$

which is equivalent to Eq. (2.18).

A.2 Different predictors

In this section, we explain different travel time predictors which are compared with the proposed method.

A.2.1 Instantaneous travel time

Instantaneous travel time is calculated based on the assumption that the current state does not change with time, i.e.,

$$
v\left(t',x\right)=v\left(t,x\right),\ \forall t'>t,\ \forall x, \tag{A.34}
$$

when the current time is t. The travel time $itt(t)$ is then estimated based on this velocity field with Algorithm 1.

A.2.2 k-Nearest neighbor

The k-Nearest neighbor (k-NN) method estimates an unknown velocity field with the k most similar (or nearest) days in the training set up to a current time t in terms of euclidean distance. Specifically, the velocity field for the rest of the day (after the current time) is estimated as the average of the velocities of the k nearest neighbors.. We set $k = 1$, meaning that we choose the most similar day in the training set for prediction. The travel time based on the nearest neighbor method is calculated by Algorithm 1.

A.2.3 Support vector regression

To implement a support vector regression (SVR) method, we have followed the same procedure of the previous work [30]. However, instead of using actual travel times as inputs, we put instantaneous travel times because that we don't know the actual travel time at the time of estimation. We used the past 5 instantaneous travel times, i.e., $itt(t-4)$, $itt(t-3)$, ... , $itt(t)$ as input variables. These input variables are scaled to have a zero mean and a unit variance. We set the target as the actual travel time with prediction horizon h, $a(t+h)$. Like in Ref. [30], the linear kernel was chosen with the parameter setting $C = 1000$ and $\tau = 0.1$.

A.2.4 Artificial neural network

We have designed a simple vanilla artificial neural network (ANN) for comparison using the same scaled input and target variables as in the SVR above. We set one hidden layer with 10 neurons. We used the MLPRegressor module of Scikit-learn python package and set all the parameter settings as the default setting except for the aforementioned hidden layer setting.

B Appendix for Chapter 3

B.1 Volume conservation of mixture of heat diffusion

By definition (in Eq. (3.6)), the graph Laplacian $\mathbf{L}(\mathcal{G})$ has an eigenvector $\frac{1}{\sqrt{N}}\mathbf{1}$ with the corresponding eigenvalue 0. Let the eigen-decomposition of a matrix be

$$\mathbf{L}(\mathcal{G}) = \mathbf{V}\mathbf{D}\mathbf{V}^T, \tag{B.1}$$

where an orthonormal matrix $\mathbf{V}$ and a diagonal matrix $\mathbf{D}$ contain eigenvectors and corresponding eigenvalues, respectively. Since the orthonormal matrix $\mathbf{V}$ contains the eigenvector $\frac{1}{\sqrt{N}}\mathbf{1}$,

$$\begin{aligned}\mathbf{1}^T\tilde{\mathbf{x}}^d_{t+1}(\tau) &\overset{(3.8)}{=} \mathbf{1}^T\mathbf{H}^{\mathcal{G}}(\tau)\mathbf{x}^d_t \\ &\overset{(3.5)}{=} \mathbf{1}^T e^{-\tau\mathbf{L}(\mathcal{G})}\mathbf{x}^d_t = \mathbf{1}^T\mathbf{V}e^{-\tau\mathbf{D}}\mathbf{V}^T\mathbf{x}^d_t \\ &= \frac{N}{\sqrt{N}}\frac{1}{\sqrt{N}}\mathbf{1}^T\mathbf{x}^d_t = \mathbf{1}^T\mathbf{x}^d_t.\end{aligned} \tag{B.2}$$

Therefore,

$$\begin{aligned}\mathbf{1}^T\tilde{\mathbf{x}}^d_{t+1}(\mathcal{T}) &= \mathbf{1}^T\mathbf{H}^{\mathcal{G}}(\mathcal{T})\mathbf{x}^d_t = \mathbf{1}^T\left(\sum_\tau \pi^{(\tau)}\mathbf{H}^{\mathcal{G}}(\tau)\right)\mathbf{x}^d_t \\ &= \sum_\tau \pi^{(\tau)}\mathbf{1}^T\mathbf{H}^{\mathcal{G}}(\tau)\mathbf{x}^d_t = \sum_\tau \pi^{(\tau)}\mathbf{1}^T\mathbf{x}^d_t \\ &= \mathbf{1}^T\mathbf{x}^d_t\sum_\tau \pi^{(\tau)} = \mathbf{1}^T\mathbf{x}^d_t.\end{aligned} \tag{B.3}$$

B.2 Evidence

$$
\begin{aligned}
& f(\mathbf{X}_{t+1}|\mathbf{X}_t,\alpha_t,\Pi_t) \\
&= \int f(\mathbf{X}_{t+1}|\mathbf{X}_t,\mathbf{H}_t,\alpha_t) f(\mathbf{H}_t|\Pi_t) d\mathbf{H}_t \\
&\propto \int e^{-\frac{1}{2}\alpha_t \mathrm{tr}\{(\mathbf{X}_{t+1}-\mathbf{H}_t\mathbf{X}_t)(\mathbf{X}_{t+1}-\mathbf{H}_t\mathbf{X}_t)^T\}} \\
&\qquad \cdot e^{-\frac{1}{2}\gamma_t \mathrm{tr}\{(\mathbf{H}_t-\mathbf{H}_t^{\mathscr{G}}(\mathscr{T}))(\mathbf{H}_t-\mathbf{H}_t^{\mathscr{G}}(\mathscr{T}))^T\}} d\mathbf{H}_t \\
&\propto e^{-\frac{1}{2}\alpha_t\left(\mathbf{X}_{t+1}(\mathbf{I}-\alpha_t\mathbf{X}_t^T\boldsymbol{\Sigma}_t\mathbf{X}_t)\mathbf{X}_{t+1}^T - 2\gamma_t\mathbf{H}_t^{\mathscr{G}}(\mathscr{T})\boldsymbol{\Sigma}_t\mathbf{X}_t\mathbf{X}_{t+1}^T\right)} \\
&\qquad \cdot \int (2\pi)^{-\frac{N^2}{2}} |\boldsymbol{\Sigma}_t|^{-\frac{N}{2}} e^{-\frac{1}{2}\mathrm{tr}\{(\mathbf{H}_t-\hat{\mathbf{H}}_t)\boldsymbol{\Sigma}_t^{-1}(\mathbf{H}_t-\hat{\mathbf{H}}_t)^T\}} d\mathbf{H}_t \\
&\propto e^{-\frac{1}{2}\alpha_t\left(\mathbf{X}_{t+1}(\mathbf{I}-\alpha_t\mathbf{X}_t^T\boldsymbol{\Sigma}_t\mathbf{X}_t)\mathbf{X}_{t+1}^T - 2\gamma_t\mathbf{H}_t^{\mathscr{G}}(\mathscr{T})\boldsymbol{\Sigma}_t\mathbf{X}_t\mathbf{X}_{t+1}^T\right)} \\
&\propto e^{-\frac{1}{2}\mathrm{tr}\{\alpha_t(\mathbf{X}_{t+1}-\mathbf{H}_t^{\mathscr{G}}(\mathscr{T})\mathbf{X}_t)(\mathbf{I}+\alpha_t\gamma_t^{-1}\mathbf{X}_t^T\mathbf{X}_t)^{-1}(\mathbf{X}_{t+1}-\mathbf{H}_t^{\mathscr{G}}(\mathscr{T})\mathbf{X}_t)^T\}},
\end{aligned}
\tag{B.4}
$$

where $\boldsymbol{\Sigma}_t^{-1} = \alpha_t\mathbf{X}_t\mathbf{X}_t^T + \gamma_t\mathbf{I}$.

B.3 Posterior distribution

When $h = 1$,

$$
\begin{aligned}
& f(\mathbf{x}_{t+1}|\mathbf{x}_t,\mathbf{X}_{t+1},\mathbf{X}_t) \\
&= \int f(\mathbf{x}_{t+1}|\mathbf{x}_t,\mathbf{H}_t,\alpha_t) f(\mathbf{H}_t|\mathbf{X}_{t+1},\mathbf{X}_t,\alpha_t,\gamma_t,\Pi_t,\mathscr{G}) dH_t \\
&= f(\mathbf{x}_{t+1}|\mathbf{x}_t,\hat{\mathbf{H}}_t,\alpha_t) = \mathscr{N}(\hat{\mathbf{H}}_t\mathbf{x}_t,\alpha_t^{-1}\mathbf{I}).
\end{aligned}
\tag{B.5}
$$

Assume the statement is true for $h = l-1$ so that

$$
f(\mathbf{x}_{t+l-1}|\mathbf{x}_t,\mathbf{X}_{t:t+l-1}) = \mathscr{N}(\hat{\mathbf{H}}_{t+l-2\leftarrow t}\mathbf{x}_t,\mathbf{R}_{t+l-2}),
\tag{B.6}
$$

where $\hat{\mathbf{H}}_{t+l-2\leftarrow t} = \hat{\mathbf{H}}_{t+l-2}\hat{\mathbf{H}}_{t+l-3}\cdots\hat{\mathbf{H}}_t$. By the chain rule,

$$
\begin{aligned}
& f(\mathbf{x}_{t+l}|\mathbf{x}_t,\mathbf{X}_{t:t+l}) \\
&= \int f(\mathbf{x}_{t+l}|\mathbf{x}_{t+l-1},\mathbf{X}_{t+l-1}) f(\mathbf{x}_{t+l-1}|\mathbf{x}_t,\mathbf{X}_{t:t+l-1}) d\mathbf{x}_{t+l-1}.
\end{aligned}
\tag{B.7}
$$

Since

$$
\begin{aligned}
& f(\mathbf{x}_{t+l}|\mathbf{x}_{t+l-1},\mathbf{X}_{t+l},\mathbf{X}_{t+l-1}) \\
&\qquad\qquad \overset{(B.5)}{=} \mathscr{N}(\hat{\mathbf{H}}_{t+l-1}\mathbf{x}_{t+l-1},\alpha_{t+l-1}^{-1}\mathbf{I}), \\
& f(\mathbf{x}_{t+l-1}|\mathbf{x}_t,\mathbf{X}_{t+l-1},\cdots,\mathbf{X}_t) \overset{(B.6)}{=} \mathscr{N}(\hat{\mathbf{H}}_{t+l-2\leftarrow t}\mathbf{x}_t,\mathbf{R}_{t+l-2}),
\end{aligned}
\tag{B.8}
$$

$$
\begin{aligned}
& f(\mathbf{x}_{t+l}|\mathbf{x}_t, \mathbf{X}_{t+l}, \cdots, \mathbf{X}_t) \\
&= \int \mathcal{N}(\hat{\mathbf{H}}_{t+l-1}\mathbf{x}_{t+l-1}, \alpha_{t+l-1}^{-1}\mathbf{I}) \\
&\qquad \cdot \mathcal{N}(\hat{\mathbf{H}}_{t+l-2\leftarrow t}\mathbf{x}_t, \mathbf{R}_{t+l-2}) d\mathbf{x}_{t+l-1} \\
&\propto \int exp\Big(-\frac{1}{2}\{\alpha_{t+l-1}(\mathbf{x}_{t+l} - \hat{\mathbf{H}}_{t+l-1}\mathbf{x}_{t+l-1})^T \\
&\qquad \cdot (\mathbf{x}_{t+l} - \hat{\mathbf{H}}_{t+l-1}\mathbf{x}_{t+l-1}) \\
&+ (\mathbf{x}_{t+l-1} - \hat{\mathbf{H}}_{t+l-2\leftarrow t}\mathbf{x}_t)^T \mathbf{R}_{t+l-2} \\
&\qquad \cdot (\mathbf{x}_{t+l-1} - \hat{\mathbf{H}}_{t+l-2\leftarrow t}\mathbf{x}_t)\}\Big) d\mathbf{x}_{t+l-1} \\
&\propto exp\Big(-\frac{1}{2}\big(\alpha_{t+l-1}\mathbf{x}_{t+l}^T\mathbf{x}_{t+l} \\
&\qquad - (\alpha_{t+l-1}\hat{\mathbf{H}}_{t+l-1}^T\mathbf{x}_{t+l} + \mathbf{R}_{t+l-2}^{-1}\hat{\mathbf{H}}_{t+l-2\leftarrow t}\mathbf{x}_t)^T \\
&\qquad \cdot (\alpha_{t+l-1}\hat{\mathbf{H}}_{t+l-1}^T\hat{\mathbf{H}}_{t+l-1} + \mathbf{R}_{t+l-2}^{-1})^{-1} \\
&\qquad \cdot (\alpha_{t+l-1}\hat{\mathbf{H}}_{t+l-1}^T\mathbf{x}_{t+l} + \mathbf{R}_{t+l-2}^{-1}\hat{\mathbf{H}}_{t+l-2\leftarrow t}\mathbf{x}_t)\big)\Big) \\
&\propto exp\Big(-\frac{1}{2}\alpha_{t+l-1} \\
&\cdot \big(\mathbf{x}_{t+l}^T(\mathbf{I} - \alpha_{t+l-1}\hat{\mathbf{H}}_{t+l-1}(\alpha_{t+l-1}\hat{\mathbf{H}}_{t+l-1}^T\hat{\mathbf{H}}_{t+l-1} \\
&\qquad + \mathbf{R}_{t+l-2}^{-1})^{-1}\hat{\mathbf{H}}_{t+l-1}^T)\mathbf{x}_{t+l} \\
&- 2\mathbf{x}_{t+l}^T\hat{\mathbf{H}}_{t+l-1}(\alpha_{t+l-1}\hat{\mathbf{H}}_{t+l-1}^T\hat{\mathbf{H}}_{t+l-1} + \mathbf{R}_{t+l-2}^{-1})^{-1} \\
&\qquad \cdot \mathbf{R}_{t+l-2}^{-1}\hat{\mathbf{H}}_{t+l-2\leftarrow t}\mathbf{x}_t\big)\Big).
\end{aligned}
\tag{B.9}
$$

Applying matrix inversion lemma, Eq. (B.9) becomes

$$
\begin{aligned}
& exp\Big(-\frac{1}{2}\alpha_{t+l-1} \\
&\cdot \big(\mathbf{x}_{t+l}^T(\mathbf{I} + \alpha_{t+l-1}\hat{\mathbf{H}}_{t+l-1}\mathbf{R}_{t+l-2}\hat{\mathbf{H}}_{t+l-1}^T)^{-1}\mathbf{x}_{t+l} \\
&- 2\mathbf{x}_{t+l}^T(\mathbf{I} + \alpha_{t+l-1}\hat{\mathbf{H}}_{t+l-1}\mathbf{R}_{t+l-2}\hat{\mathbf{H}}_{t+l-1}^T)^{-1} \\
&\qquad \cdot \hat{\mathbf{H}}_{t+l-1}\hat{\mathbf{H}}_{t+l-2\leftarrow t}\mathbf{x}_t\big)\Big) \\
&\propto exp\Big(-\frac{1}{2}(\mathbf{x}_{t+l} - \hat{\mathbf{H}}_{t+l-1}\hat{\mathbf{H}}_{t+l-2\leftarrow t}\mathbf{x}_t)^T \\
&\qquad \cdot \mathbf{R}_{t+l-1}^{-1}(\mathbf{x}_{t+l} - \hat{\mathbf{H}}_{t+l-1}\hat{\mathbf{H}}_{t+l-2\leftarrow t}\mathbf{x}_t)\Big),
\end{aligned}
\tag{B.10}
$$

where $\mathbf{R}_{t+l-1} = \alpha_{t+l-1}^{-1}\mathbf{I} + \hat{\mathbf{H}}_{t+l-1}\mathbf{R}_{t+l-2}\hat{\mathbf{H}}_{t+l-1}^T$ and by definition $\hat{\mathbf{H}}_{t+l-1\leftarrow t} = \hat{\mathbf{H}}_{t+l-1}\hat{\mathbf{H}}_{t+l-2\leftarrow t}$, so

$$
f(\mathbf{x}_{t+l}|\mathbf{x}_t, \mathbf{X}_{t+l}, \cdots, \mathbf{x}_t) = \mathcal{N}(\hat{\mathbf{H}}_{t+l-1\leftarrow t}\mathbf{x}_t, \mathbf{R}_{t+l-1}). \tag{B.11}
$$

Finally, $\mathbf{x}_{t+h|t} = \hat{\mathbf{H}}_{t+h-1}\cdots\hat{\mathbf{H}}_t\mathbf{x}_t$.

Bibliography

[1] M. A. Quddus, W. Y. Ochieng, and R. B. Noland, "Current map-matching algorithms for transport applications: state-of-the art and future research directions", *Transportation research part c: Emerging technologies*, vol. 15, no. 5, pp. 312–328, 2007.

[2] E. I. Vlahogianni, J. C. Golias, and M. G. Karlaftis, "Short-term traffic forecasting: overview of objectives and methods", *Transport reviews*, vol. 24, no. 5, pp. 533–557, 2004.

[3] S. R. Chandra and H. Al-Deek, "Predictions of freeway traffic speeds and volumes using vector autoregressive models", *Journal of Intelligent Transportation Systems*, vol. 13, no. 2, pp. 53–72, 2009.

[4] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains", *IEEE signal processing magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[5] D. J. MacKay, "Bayesian interpolation", *Neural computation*, vol. 4, no. 3, pp. 415–447, 1992.

[6] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities", *Proceedings of the national academy of sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[7] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "Wavenet: a generative model for raw audio.", *SSW*, vol. 125, p. 2, 2016.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need", *Advances in neural information processing systems*, vol. 30, 2017.

[9] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, *et al.*, "Relational inductive biases, deep learning, and graph networks", *arXiv preprint arXiv:1806.01261*, 2018.

[10] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks", *arXiv preprint arXiv:1710.10903*, 2017.

[11] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks: a comprehensive review", *Computational Social Networks*, vol. 6, no. 1, pp. 1–23, 2019.

[12] M. Bai, Y. Lin, M. Ma, and P. Wang, "Travel-time prediction methods: a review", in *Smart Computing and Communication*, M. Qiu, Ed., Cham: Springer International Publishing, 2018, pp. 67–77, ISBN: 978-3-030-05755-8.

[13] M. Ben-Akiva, M. Bierlaire, D. Burton, H. N. Koutsopoulos, and R. Mishalani, "Network state estimation and prediction for real-time traffic management", *Networks and spatial economics*, vol. 1, no. 3-4, pp. 293–318, 2001.

[14] N. Wan, G. Gomes, A. Vahidi, and R. Horowitz, "Prediction on travel-time distribution for freeways using online expectation maximization algorithm", in *Transportation Research Board 93rd Annual Meeting*, 2014.

[15] S. Takaba, T. Morita, T. Hada, T. Usami, and M. Yamaguchi, "Estimation and measurement of travel time by vehicle detectors and license plate readers", in *Vehicle Navigation and Information Systems Conference, 1991*, IEEE, vol. 2, 1991, pp. 257–267.

[16] A. Skabardonis and N. Geroliminis, "Real-time estimation of travel times on signalized arterials", in *International Symposium on Transportation and Traffic Theory (ISTTT)*, Elsevier, 2005, pp. 387–406.

[17] C. Nanthawichit, T. Nakatsuji, and H. Suzuki, "Application of probe-vehicle data for real-time traffic-state estimation and short-term travel-time prediction on a freeway", *Transportation Research Record: Journal of the Transportation Research Board*, no. 1855, pp. 49–59, 2003.

[18] J. Rice and E. Van Zwet, "A simple and effective method for predicting travel times on freeways", in *Intelligent Transportation Systems, 2001. Proceedings. 2001 IEEE*, IEEE, 2001, pp. 227–232.

[19] X. Zhang and J. A. Rice, "Short-term travel time prediction", *Transportation Research Part C: Emerging Technologies*, vol. 11, no. 3-4, pp. 187–210, 2003.

[20] M. Yang, Y. Liu, and Z. You, "The reliability of travel time forecasting", *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 1, pp. 162–171, 2009.

[21] J. Xia, M. Chen, and W. Huang, "A multistep corridor travel-time prediction method using presence-type vehicle detector data", *Journal of Intelligent Transportation Systems*, vol. 15, no. 2, pp. 104–113, 2011.

[22] 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*, vol. 16, no. 2, pp. 719–729, 2014.

[23] A. Salamanis, D. D. Kehagias, C. K. Filelis-Papadopoulos, D. Tzovaras, and G. A. Gravvanis, "Managing spatial graph dependencies in large volumes of traffic data for travel-time prediction", *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1678–1687, 2015.

[24] S. I.-J. Chien and C. M. Kuchipudi, "Dynamic Travel Time Prediction with Real-Time and Historic Data", *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 608–616, Nov. 2003.

[25] C. Kuchipudi and S. Chien, "Development of a hybrid model for dynamic travel-time prediction", *Transportation Research Record: Journal of the Transportation Research Board*, no. 1855, pp. 22–31, 2003.

[26] J. Van Lint, "Online learning solutions for freeway travel time prediction", *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 1, pp. 38–47, 2008.

[27] A. Achar, D. Bharathi, B. A. Kumar, and L. Vanajakshi, "Bus arrival time prediction: a spatial kalman filter approach", *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[28] C. I. van Hinsbergen, J. Van Lint, and H. Van Zuylen, "Bayesian committee of neural networks to predict travel times with confidence intervals", *Transportation Research Part C: Emerging Technologies*, vol. 17, no. 5, pp. 498–509, 2009.

[29] X. Fei, C.-C. Lu, and K. Liu, "A bayesian dynamic linear model approach for real-time short-term freeway travel time prediction", *Transportation Research Part C: Emerging Technologies*, vol. 19, no. 6, pp. 1306–1318, 2011.

[30] C.-H. Wu, C.-C. Wei, D.-C. Su, M.-H. Chang, and J.-M. Ho, "Travel time prediction with support vector regression", in *Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE*, IEEE, vol. 2, 2003, pp. 1438–1442.

[31] M. Castro-Neto, Y.-S. Jeong, M.-K. Jeong, and L. D. Han, "Online-svr for short-term traffic flow prediction under typical and atypical traffic conditions", *Expert systems with applications*, vol. 36, no. 3, pp. 6164–6173, 2009.

[32] P. Gao, J. Hu, H. Zhou, and Y. Zhang, "Travel time prediction with immune genetic algorithm and support vector regression", in *2016 12th World Congress on Intelligent Control and Automation (WCICA)*, IEEE, 2016, pp. 987–992.

[33] D. Park and L. R. Rilett, "Forecasting Freeway Link Travel Times with a Multilayer Feedforward Neural Network", *Computer-Aided Civil and Infrastructure Engineering*, vol. 14, no. 5, pp. 357–367, Sep. 1999.

[34] H. Dia, "An object-oriented neural network approach to short-term traffic forecasting", *European Journal of Operational Research*, vol. 131, no. 2, pp. 253–261, 2001.

[35] A. Dharia and H. Adeli, "Neural network model for rapid forecasting of freeway link travel time", *Engineering Applications of Artificial Intelligence*, vol. 16, no. 7-8, pp. 607–613, 2003.

[36] S. Innamaa, "Short-term prediction of travel time using neural networks on an interurban highway", *Transportation*, vol. 32, no. 6, pp. 649–669, 2005.

[37] J. Van Lint, S. Hoogendoorn, and H. J. van Zuylen, "Accurate freeway travel time prediction with state-space neural networks under missing data", *Transportation Research Part C: Emerging Technologies*, vol. 13, no. 5-6, pp. 347–369, 2005.

[38] H. Liu, H. Van Zuylen, H. Van Lint, and M. Salomons, "Predicting urban arterial travel time with state-space neural networks and kalman filters", *Transportation Research Record*, vol. 1968, no. 1, pp. 99–108, 2006.

[39] C.-S. Li and M.-C. Chen, "A data mining based approach for travel time prediction in freeway with non-recurrent congestion", *Neurocomputing*, vol. 133, pp. 74–83, 2014.

[40] Y. Hou and P. Edara, "Network scale travel time prediction using deep learning", *Transportation Research Record*, vol. 2672, no. 45, pp. 115–123, 2018.

[41] Y. Duan, Y. Lv, and F.-Y. Wang, "Travel time prediction with lstm neural network", in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2016, pp. 1053–1058.

[42] Y. Liu, Y. Wang, X. Yang, and L. Zhang, "Short-term travel time prediction by deep learning: a comparison of different lstm-dnn models", in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2017, pp. 1–8.

[43] B. Hamner, "Predicting travel times with context-dependent random forests by modeling local and aggregate traffic flow", in *2010 IEEE International Conference on Data Mining Workshops*, IEEE, 2010, pp. 1357–1359.

[44] Y. Zhang and A. Haghani, "A gradient boosting method to improve travel time prediction", *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 308–324, 2015.

[45] N. C. Petersen, F. Rodrigues, and F. C. Pereira, "Multi-output bus travel time prediction with convolutional lstm neural network", *Expert Systems with Applications*, vol. 120, pp. 426–435, 2019.

[46] S. Robinson and J. W. Polak, "Modeling urban link travel time with inductive loop detector data by using the k-nn method", *Transportation research record*, vol. 1935, no. 1, pp. 47–56, 2005.

[47] M. Yildirimoglu and N. Geroliminis, "Experienced travel time prediction for congested freeways", *Transportation Research Part B: Methodological*, vol. 53, pp. 45–63, Jul. 2013.

[48] T. Kailath, B. Hassidi, and A. H. Sayed, *Linear estimation*. Prentice-Hall, 2000.

[49] W. Feller, *An introduction to probability theory and its applications*. John Wiley & Sons, 2008, vol. 2.

[50] R. Wan, S. Mei, J. Wang, M. Liu, and F. Yang, "Multivariate temporal convolutional network: a deep neural networks approach for multivariate time series forecasting", *Electronics*, vol. 8, no. 8, 2019, ISSN: 2079-9292. DOI: 10.3390/electronics8080876. [Online]. Available: https://www.mdpi.com/2079-9292/8/8/876.

[51] M. Das and S. K. Ghosh, "Sembnet: a semantic bayesian network for multivariate prediction of meteorological time series data", *Pattern Recognition Letters*, vol. 93, pp. 192–201, 2017.

[52] T. Ouyang, X. Zha, and L. Qin, "A combined multivariate model for wind power prediction", *Energy Conversion and Management*, vol. 144, pp. 361–373, 2017.

[53] K. Wang, K. Li, L. Zhou, Y. Hu, Z. Cheng, J. Liu, and C. Chen, "Multiple convolutional neural networks for multivariate time series prediction", *Neurocomputing*, vol. 360, pp. 107–119, 2019.

[54] S. Huang, D. Wang, X. Wu, and A. Tang, "Dsanet: dual self-attention network for multivariate time series forecasting", in *Proceedings of the 28th ACM international conference on information and knowledge management*, 2019, pp. 2129–2132.

[55] T. Mai, B. Ghosh, and S. Wilson, "Multivariate short-term traffic flow forecasting using bayesian vector autoregressive moving average model", Tech. Rep. 12-3728, 2012.

[56] S. Kwak and N. Geroliminis, "Travel time prediction for congested freeways with a dynamic linear model", *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[57] L. Cavalcante, R. J. Bessa, M. Reis, and J. Browell, "Lasso vector autoregression structures for very short-term wind power forecasting", *Wind Energy*, vol. 20, no. 4, pp. 657–675, 2017.

[58] W. B. Nicholson, I. Wilms, J. Bien, and D. S. Matteson, "High dimensional forecasting via interpretable vector autoregression", *Journal of Machine Learning Research*, vol. 21, no. 166, pp. 1–52, 2020.

[59] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: data-driven traffic forecasting", in *International Conference on Learning Representations (ICLR '18)*, 2018.

[60] Z. Cui, K. Henrickson, R. Ke, and Y. Wang, "Traffic graph convolutional recurrent neural network: a deep learning framework for network-scale traffic learning and forecasting", *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[61] C. Chen, K. Li, S. G. Teo, X. Zou, K. Wang, J. Wang, and Z. Zeng, "Gated residual recurrent graph neural networks for traffic prediction", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 485–492.

[62] C. Zhang, J. James, and Y. Liu, "Spatial-temporal graph attention networks: a deep learning approach for traffic forecasting", *IEEE Access*, vol. 7, pp. 166 246–166 256, 2019.

[63] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting", in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 3634–3640.

[64] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-gcn: a temporal graph convolutional network for traffic prediction", *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[65] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling", in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, AAAI Press, 2019, pp. 1907–1913.

[66] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: multivariate time series forecasting with graph neural networks", in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 753–763.

[67] B. Xu, H. Shen, Q. Cao, Y. Qiu, and X. Cheng, "Graph wavelet neural network", in *International Conference on Learning Representations*, 2018.

[68] S. Du, T. Li, Y. Yang, and S.-J. Horng, "Multivariate time series forecasting via attention-based encoder–decoder framework", *Neurocomputing*, vol. 388, pp. 269–279, 2020.

[69] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, "Deep multi-view spatial-temporal network for taxi demand prediction", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[70] L. Munkhdalai, T. Munkhdalai, K. H. Park, T. Amarbayasgalan, E. Batbaatar, H. W. Park, and K. H. Ryu, "An end-to-end adaptive input selection with dynamic weights for forecasting multivariate time series", *IEEE Access*, vol. 7, pp. 99 099–99 114, 2019.

[71] J. Du Preez and S. F. Witt, "Univariate versus multivariate time series forecasting: an application to international tourism demand", *International Journal of Forecasting*, vol. 19, no. 3, pp. 435–451, 2003.

[72] D. Helbing, "Traffic and related self-driven many-particle systems", *Reviews of modern physics*, vol. 73, no. 4, p. 1067, 2001.

[73] R. I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete structures", in *Proceedings of the 19th international conference on machine learning*, vol. 2002, 2002, pp. 315–22.

[74] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization", *SIAM Journal on scientific computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[75] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[76] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, "Urban traffic prediction from spatio-temporal data using deep meta learning", in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1720–1730.

[77] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction", *Sensors*, vol. 17, no. 4, p. 818, 2017.

[78] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks", *Sensors*, vol. 17, no. 7, p. 1501, 2017.

[79] Y. Zhang, S. Wang, B. Chen, J. Cao, and Z. Huang, "Trafficgan: network-scale deep traffic prediction with generative adversarial nets", *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[80] Z. He, C.-Y. Chow, and J.-D. Zhang, "Stcnn: a spatio-temporal convolutional neural network for long-term traffic prediction", in *2019 20th IEEE International Conference on Mobile Data Management (MDM)*, IEEE, 2019, pp. 226–233.

[81] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: data-driven traffic forecasting", in *International Conference on Learning Representations (ICLR '18)*, 2018.

[82] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling", *arXiv preprint arXiv:1906.00121*, 2019.

[83] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting", *arXiv preprint arXiv:1709.04875*, 2017.

[84] S. Kwak, N. Geroliminis, and P. Frossard, "Traffic signal prediction on transportation networks using spatio-temporal correlations on graphs", *IEEE Transactions on Signal and Information Processing over Networks*, vol. 7, pp. 648–659, 2021.

[85] C. Daganzo, *Fundamentals of Transportation and Traffic Operations*. Emerald Group Publishing Limited, 1997, ISBN: 9780080427850.

[86] J. Shi and J. Malik, "Normalized cuts and image segmentation", *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[87] U. Von Luxburg, "A tutorial on spectral clustering", *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[88] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks", in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[89] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: encoder-decoder approaches", *arXiv preprint arXiv:1409.1259*, 2014.

[90] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks", *arXiv preprint arXiv:1506.03099*, 2015.

[91] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[92] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU Press, 2012, vol. 3.

Semin Kwak

semin.kwak@epfl.ch | semink.github.io

Education

École Polytechnique Fèdèrale de Lausanne (EPFL) — Lausanne, Switzerland
Ph.D. in Electrical engineering — *Apr. 2017 – Sep. 2022 (expected)*

Korea Advanced Institute of Science and Technology (KAIST) — Daejeon, Republic of Korea
M.S. in Electrical engineering — *Mar. 2013 – Feb. 2015*

Yonsei University — Seoul, Republic of Korea
B.S. in Electrical engineering — *Mar. 2007 – Feb. 2013*

Research Experience

Research Assistant — Apr. 2017 – Present
EPFL — *Lausanne, Switzerland*

- Developing a traffic prediction method for a large transportation sensor networks (related publication: [J1], ongoing works: [O2, O3])
- Developing tools for analyzing citation graph in transportation field (ongoing work: [O1])
- Developed a traffic/travel time prediction method for congested freeways (related publication: [J2])

Research Intern — May 2016 – Mar. 2017
Korea Institute of Science and Technology (KIST) — *Seoul, Republic of Korea*

- Worked in the Center for Robotics Research which engages in research on human-robot interaction and machine learning
- Developed a Support Vector Machine (SVM) based classifier for emergency calls

Research Assistant — Mar. 2013 – Feb. 2015
KAIST — *Daejeon, Republic of Korea*

- Worked in the Scientific Computing Laboratory (SCLAB) whose researches mainly focus on estimation/detection theory, radar signal processing and communication
- Studied convex optimization-based approaches for beam pattern synthesis (related publications: [J5, J3])
- Applied the compressive sensing technique to beam pattern synthesis to make a sparse array (related publication: [J4])
- Collaborated with Samsung Electronics about a beamforming method for serving multiple users in the 5G communication system (related patent: [P1])

Undergraduate Research Assistant — June 2012 – Feb. 2013
Yonsei University — *Seoul, Republic of Korea*

- Developed a method to sharpen Medical Resonance Imaging (MRI) images exploiting the Constrained Least-Squares (CLS) filter

Teaching Experience

Teaching Assistance — Sep. 2017 – Sep. 2020
EPFL — *Lausanne, Switzerland*

- CIVIL457 Fundamentals of Traffic Operations and Control

Undergraduate tutoring program — Sep. 2013 – Feb. 2014
KAIST — *Daejeon, Republic of Korea*

- EE202 Signals and Systems

Teaching Assistance — Mar. 2014 – Dec. 2014
KAIST — *Daejeon, Republic of Korea*

- EE505 Electronics Design Laboratory
- EE746 Radar Systems

Undergraduate tutoring program — Sep. 2012 – Dec. 2012
Yonsei University — *Seoul, Republic of Korea*

- EEE3440 Digital Communication

Mentoring program Sep. 2011 – Jan. 2012
Korea JoongAng Daily *Seoul, Republic of Korea*
- High school mathematics

Projects

The Walking Data | *Javascript, HTML/CSS, D3.js* Sep. 2018 – Dec. 2018
- Developed entire 2D visualization/functions of the project

Awards

Korea Government Fellowship for Graduate Study Mar. 2013 – Feb. 2015
- Full supports from the Korean government including the tuition and stipend

National Science and Engineering Undergraduate Scholarship Mar. 2007 – Feb. 2013
- Full scholarships from the Korean government by the excellence of academic work

Technical Skills

Languages: Python, C/C++, MATLAB, JavaScript, HTML/CSS
Developer Tools: Git, Docker, PyCharm, Eclipse
Libraries: Pandas, NumPy, Matplotlib, Plotly, Pytorch, Pytorch-lightning, Tensorflow, Keras, Scikit-learn

Publications

[J1] **Semin Kwak**, Nikolas Geroliminis, and Pascal Frossard. "Traffic signal prediction on transportation networks using spatio-temporal correlations on graphs". In: *IEEE Transactions on Signal and Information Processing over Networks* 7 (2021), pp. 648–659.

[J2] **Semin Kwak** and Nikolas Geroliminis. "Travel time prediction for congested freeways with a dynamic linear model". In: *IEEE Transactions on Intelligent Transportation Systems* (2020).

[J3] **Semin Kwak**, Joohwan Chun, and Sung Hyuck Ye. "Monopulse beam synthesis using a sparse single layer of weights". In: *IEEE Transactions on Antennas and Propagation* 67.4 (2019), pp. 2787–2791.

[J4] Jaehyun Park, Jihye Lee, Joohwan Chun, and **Semin Kwak**. "Robust monopulse beam synthesis with sparse elements in linear and planar arrays with element failure detection". In: *IET Radar, Sonar & Navigation* 11.8 (2017), pp. 1251–1258.

[J5] **Semin Kwak**, Joohwan Chun, Dongmin Park, Young Kwan Ko, and Byung Lae Cho. "Asymmetric sum and difference beam pattern synthesis with a common weight vector". In: *IEEE Antennas and Wireless Propagation Letters* 15 (2016), pp. 1622–1625.

Ongoing works

[O1] Nikolas Geroliminis, **Semin Kwak**, and Fred Mannering. "Citation analysis in transportation field". (expected). Apr. 2022.

[O2] **Semin Kwak**, Nikolas Geroliminis, and Pascal Frossard. "Spatio-temporal multiresolution with Dilated Graph Convolutional neural network for traffic forecasting". (expected). June 2022.

[O3] **Semin Kwak**, Danya Li, and Nikolas Geroliminis. "TwoResNet: Two-level resolution neural network for traffic forecasting of freeway networks". (expected). 2022.

Conferences

[C1] Danya Li, **Semin Kwak**, and Nikolaos Geroliminis. "TwoResNet: Two-level resolution neural network for traffic forecasting of freeway networks". In: (Macau, China, Oct. 8–12, 2022). (submitted). IEEE ITSC 2022 - 25th IEEE International Conference on Intelligent Transportation Systems, 2022, 2022.

[C2] **Semin Kwak** and Nikolaos Geroliminis. "Data-driven traffic and experienced travel time forecasting method for freeways with a localized linear regression model". In: (Washington DC, USA, Jan. 13–17, 2019). Transportation Research Board 97th Annual Meeting, 2019. URL: http://infoscience.epfl.ch/record/264434.

[C3] **Semin Kwak** and Nikolaos Geroliminis. "Data-driven prediction of experienced travel times for freeways". In: (Athens, Greece, Sept. 5–7, 2018). hEART 2018 – 7th Symposium of the European Association for Research in Transportation, 2018. URL: http://infoscience.epfl.ch/record/264432.

Patents

[P1] **Semin Kwak**, Yong-hoon Kim, Hee-seong Yang, Sang-hyouk Choi, and Joo-hwan Chun. *Beamforming method and apparatus for serving multiple users*. US Patent 9,634,750. Apr. 2017.