Reliability-based journey time prediction via two-stream deep learning with multi-source data

Li Zhuang[a,b] and Xinyue Wu[b] and Andy H.F. Chow[b] and Wei Ma[c] and William H.K. Lam[c] and S.C. Wong[d]

[a]School of Cyber Science and Engineering, Southeast University, Nanjing, China; [b]Systems Engineering, City University of Hong Kong, Kowloon Tong, Hong Kong SAR, China; [c]Civil and Environmental Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong SAR, China;[d]Civil Engineering, The University of Hong Kong, Pokfulam, Hong Kong SAR, China

ARTICLE HISTORY
Compiled January 4, 2024

ABSTRACT
This paper presents a distribution-free reliability-based prediction approach for estimating journey time intervals with multi-source data using a two-stream deep learning framework. The prediction framework consists of a long short-term memory (LSTM) module for extracting temporal features and a convolutional neural network (CNN) module for extracting spatial-temporal features from the heterogeneous data. The precision and reliability of the prediction are assessed respectively by the Mean Prediction Interval Width (MPIW) and Prediction Interval Coverage Probability (PICP) metrics. For computational effectiveness, a Gaussian approximation is adopted to formulate a smooth and differentiable loss function for training the prediction framework. The computational experiments are conducted based on a real-world Hong Kong corridor, where multi-source data including traffic and weather conditions are collected. The proposed framework shows significant improvements over existing methods in terms of both precision and reliability over a range of traffic and weather conditions. This study contributes to the development of reliability-based intelligent transportation systems with advanced deep learning techniques.

KEYWORDS
vehicle journey times; interval prediction; traffic data fusion; deep learning; Gaussian approximation

1. Introduction

Precise and reliable prediction of vehicle journey times and their associated variability are important for the development of road performance measures and intelligent transportation systems (Chen et al., 2021; Chow et al., 2014; Qiao et al., 2016; Wu et al., 2016; Zhan et al., 2020). This need leads to a number of related studies reported in the literature. The journey time prediction algorithms found in the literature can generally be classified into two groups: model-based and data-driven (Shi et al., 2021; Yang et al., 2023). Model-based methods operate upon an underlying physical model

CONTACT Andy H.F. Chow. Email: andychow@cityu.edu.hk

of traffic flow or journey time dynamics. Balijepalli et al. (2007) present a dynamic travel time model with consideration of both within-day and day-to-day variations in a stochastic user equilibrium model. Connors and Sumalee (2009) analyze travel time variations with consideration of travel behavior in a stochastic environment. Kurzhanskiy and Varaiya (2010) present an active traffic management system for estimating and managing traffic flows and journey times based on a cell transmission model, and extend it to a model-based interval estimation framework (Kurzhanskiy and Varaiya, 2012). Sumalee et al. (2013) present a stochastic traffic flow and journey time estimation framework based on a probabilistic extension of the cell transmission model. Chow et al. (2015) develop a hybrid journey time prediction model with use of probe GPS bus data, and later apply it to bus service reliability control (Chow et al., 2017). In contrast, the data-driven methods are based on empirical data with no explicit underlying models. Billings and Yang (2006) use the autoregressive integrated moving average (ARIMA) method for urban travel time prediction. Myung et al. (2011) present a k-nearest neighbors (KNN) algorithm for travel time prediction with data inferred from an automatic toll collection system. Qiao et al. (2013) introduce a KNN-T model developed based on KNN for short-term travel time prediction. Cheng et al. (2014) present a journey time prediction algorithm with consideration of spatio-temporal characteristics of journey times using the STARIMA (Space-Time Autoregressive Integrated Moving Average) framework. Haworth et al. (2014) present a journey time prediction model with use of kernel regression. Zhang and Haghani (2015) introduce a gradient boosting regression tree for freeway journey time prediction. Woodard et al. (2017) introduce a Markov model with maximizing a posterior to predict the reliability of travel time by using GPS data in road networks. Zhong et al. (2017) adopt the functional principal component analysis for journey time prediction with abnormal road conditions. Feng et al. (2018) introduce an adaptive multi-kernel support vector machine with spatial-temporal correlations for traffic flow prediction. Chen et al. (2020) introduce a gradient boosting model for travel time prediction. Kwak and Geroliminis (2020) present a dynamic linear model for travel time prediction on congested freeways. Zhang et al. (2022) present an estimation method for travel time distributions with low-resolution video images.

Many existing traffic prediction methods require further manual refinements following the calibration process for actual implementation due to the complexity of real world traffic dynamics (Li et al., 2021a). With the advancement in data science and computing technology, we see an increasing use of deep machine learning techniques in the development of traffic estimation and prediction algorithms (Dia, 2001; Li et al., 2021a; Wang et al., 2022). Khosravi et al. (2011) and later Lin et al. (2018) present an artificial neural network (ANN) based journey time prediction framework trained by particle swarm optimization (PSO). Duan et al. (2016) adopt a long short-term memory (LSTM) based approach for journey time prediction. Zhu and Laptev (2017) present a Bayesian deep learning model with LSTM for predicting journey times for Uber services. Wu et al. (2018) present a deep neural network framework for predicting journey times along freeways. Li et al. (2019) introduce a deep feature fusion model to predict traffic speed. Yang et al. (2019) develop a parking occupancy prediction framework using LSTM and graph neural networks (GNNs). Wang et al. (2019) propose a path-based deep learning approach for traffic speed prediction in urban areas. Zhao et al. (2019) present a temporal graph convolutional network (T-GCN) model using the graph convolutional network and the gated recurrent unit for traffic prediction. Abdollahi et al. (2020) introduce a feature learning module using a deep stacked autoencoder for travel time prediction. Shi et al. (2020) introduce an attention-based

periodic-temporal neural network for traffic forecasting. Li et al. (2020) develop a two-stage journey time prediction framework with use of a deep belief network (DBN) and quantile regression. Li et al. (2022) adopt a fusion attention mechanism based LSTM for short-term traffic flow prediction. Tang et al. (2021) present a sparse denoising auto-encoders based deep architecture for citywide travel time prediction using contextual information. Yuan et al. (2021) present two convolutional neural networks (CNN) for traffic state and speed prediction. Despite the recent development on deep learning-based prediction algorithms, we have identified several issues that have not been fully addressed. Most prediction algorithms operate on a single data source and they have not been able to harness the potential benefits brought by multiple data sources. Moreover, many prediction algorithms developed in the literature are for point estimation and have not taken into account of the associated variability with interval estimates (Kurzhanskiy and Varaiya, 2012). Furthermore, the loss functions adopted in the most training process for an interval estimation algorithm are empirical and inherently non-differentiable. Such non-differentiable loss functions are difficult to optimize and hence introduce computation challenges to the training process of deep learning-based methods (Khosravi et al., 2011; Lin et al., 2018; Murphy, 2022).

This paper presents a distribution-free reliability-based prediction framework for vehicle journey times and their associated variability based on a two-stream deep learning framework with use of heterogeneous traffic data collected from multiple sources, locations, times, and days. The heterogeneous data sources (e.g. travel time, spot speed, weather) provide ranges of spatial and temporal features of prevailing traffic flow for precise and reliable journey time prediction. The spatial and temporal features in the input data are extracted through the two-stream architecture consisting of a long short-term memory (LSTM) and a convolutional neural network (CNN). To improve its computational effectiveness, this paper presents and adopts a smooth and differentiable loss function for training the reliability-based prediction framework through applying a Gaussian approximation to its inherent step loss function. It is shown that minimizing the proposed loss function can result in an accurate prediction interval for a given significant level. The training process can hence be conducted via the effective derivative-based Adam (Kingma and Ba, 2014; Murphy, 2022) stochastic gradient search instead of using the conventional but less effective metaheuristics as in many previous studies (Khosravi et al., 2011; Lin et al., 2018). The extracted features are fused and processed through multiple fully connected layers to generate a lower and upper bound of predicted journey time at the time of interest. The performance of the proposed framework is validated and tested with traffic data collected from a selected corridor in Hong Kong. To summarize, this study contributes to the development of reliability-based intelligent transportation systems with advanced deep learning techniques through delivering:

- a distribution-free interval prediction method for vehicle journey times;
- a two-stream deep learning framework for incorporating multi-source data with different spatio-temporal features;
- a smooth and differentiable loss function for prediction training with use of Gaussian approximation.

The rest of the paper is organized as follows: Section 2 presents the proposed methodology. Section 3 shows the case study with data collected from a selected Hong Kong highway corridor. Finally, Section 4 provides some concluding remarks.

2. Methodology

Figure 1 shows the overall architecture of the proposed prediction framework in which $\boldsymbol{X}_t$ represents all data available at a present time t and includes pure time series datasets $\boldsymbol{X}_t^1$ and a spatio-temporal dataset $\boldsymbol{X}_t^2$. Given the data $\boldsymbol{X}_t$, the prediction framework aims to generate the corresponding upper $\widehat{U}(z_t)$ and lower bounds $\widehat{L}(z_t)$ of the (unknown) journey time z_t to be experienced by a traveler embarking en-route at this time. Considering the differences in the orders and magnitudes between different types of data (e.g. journey times, speeds, rainfall intensities), all quantities in $\boldsymbol{X}_t$ will first be normalized with respect to their associated means and standard deviations (Ying et al., 2020). With this normalization process, the data will be converted to standard scores with mean and standard deviation equaling respectively to zero and one.

$\boldsymbol{X}_t^1$ and $\boldsymbol{X}_t^2$ are further denoted as $\boldsymbol{x}_t$ and $\mathcal{X}^{ST}$ respectively after normalization as shown in the figure. The notation $\boldsymbol{x}_t$ represents a vector containing all pure time series data (e.g. vehicle journey times and rainfall intensities over previous time intervals) and $\mathcal{X}^{ST}$ is a multi-dimensional tensor storing data with spatial and external features (e.g. previous spot speeds observed at specific locations along the route of interest and measurements on previous days). The pure time series data $\boldsymbol{x}_t$ will be fed into a temporal feature extraction stream (TFES) in the prediction framework which is constructed by a LSTM, while the multi-dimensional data in $\mathcal{X}^{ST}$ will be fed into a spatio-temporal feature extraction stream (STFES) that is built by a CNN. The CNN is a specific kind of artificial neural network designed for incorporating high-dimensional input data structure (e.g. tensors) by having its hidden layers performing convolution operation (Goodfellow et al., 2016). The extracted features from $\boldsymbol{X}_t$ through the two streams will be integrated and renamed as the comprehensive spatio-temporal features eventually to derive the corresponding upper bound $\widehat{U}(z_t)$ and lower bound $\widehat{L}(z_t)$ estimates of the vehicle journey time z_t after denormalization.

The following sections will first present the construction of the TFES and STFES, followed by the training and validation procedures. We refer the readers to Goodfellow et al. (2016) and Murphy (2022) for further details of the computational properties of the deep learning-based methods presented herein.

2.1. Temporal feature extraction stream (TFES)

The temporal feature extraction stream is developed based on the long short-term memory (LSTM), which is an extension form of the standard recurrent neural network (RNN) (Goodfellow et al., 2016; Hochreiter and Schmidhuber, 1997) designed for handling sequential and time series data (Duan et al., 2016; Li et al., 2021b; Wang et al., 2019, 2020; Yang et al., 2019). Figure 2 shows the structure of each LSTM hidden layer, known as 'cell', in the LSTM. Each LSTM cell consists of signals generated from an input gate vector $\boldsymbol{i}_t$, a forget gate vector $\boldsymbol{f}_t$, and an output gate vector $\boldsymbol{o}_t$. These signals ($\boldsymbol{i}_t$, $\boldsymbol{f}_t$, $\boldsymbol{o}_t$) govern the information flow in the LSTM cell over time t. We further have $\widetilde{\boldsymbol{m}}_t$ and $\boldsymbol{m}_t$ as vectors representing the 'memory' (Goodfellow et al., 2016; Yang et al., 2019) generated within the LSTM cell at each time t. The 'memory' vectors contain and store prevailing state conditions at different stages within the LSTM cell.

The variables in the LSTM cell evolve as follows. We first determine the vectors from the input gate, the forget gate, the output gate, and the memory cell based upon

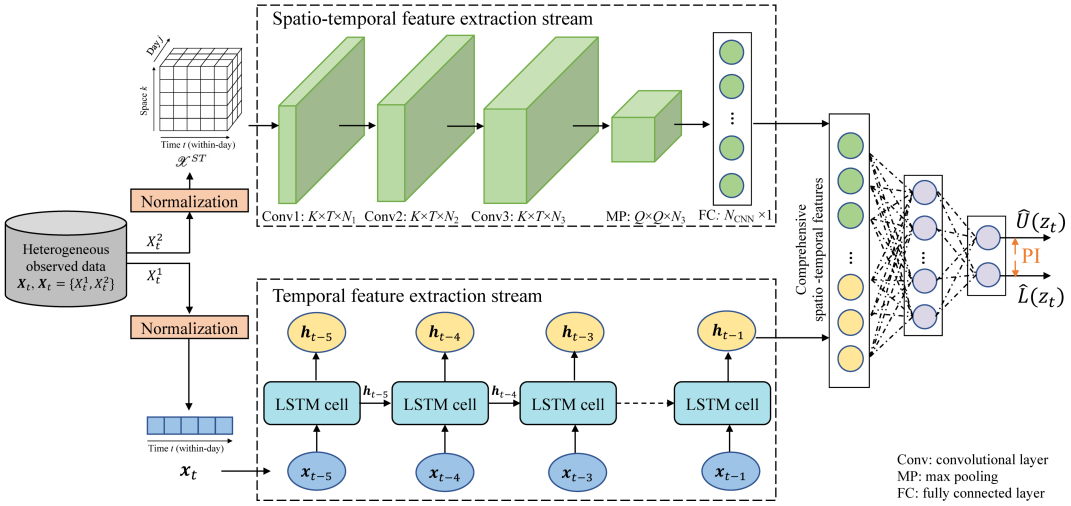


Figure 1. Architecture of the proposed journey time prediction framework

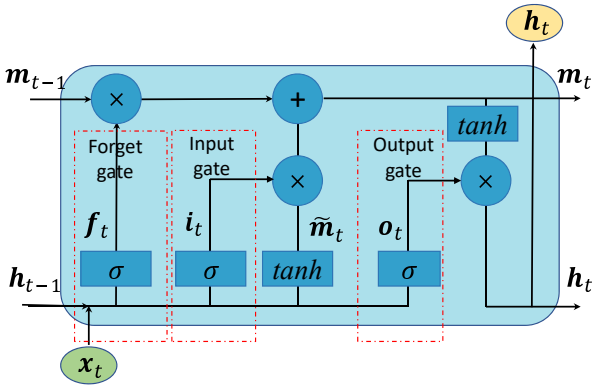


Figure 2. Structure of a basic LSTM cell

the current input $\boldsymbol{x}_t$ and previous LSTM hidden layer output $\boldsymbol{h}_{t-1}$ as:

$$\begin{aligned} \text{(input gate)} \quad & \boldsymbol{i}_t = \sigma(\boldsymbol{W}^i \boldsymbol{x}_t + \boldsymbol{U}^i \boldsymbol{h}_{t-1}) \\ \text{(forget gate)} \quad & \boldsymbol{f}_t = \sigma(\boldsymbol{W}^f \boldsymbol{x}_t + \boldsymbol{U}^f \boldsymbol{h}_{t-1}) \\ \text{(output gate)} \quad & \boldsymbol{o}_t = \sigma(\boldsymbol{W}^o \boldsymbol{x}_t + \boldsymbol{U}^o \boldsymbol{h}_{t-1}) \\ \text{(memory cell)} \quad & \widetilde{\boldsymbol{m}}_t = \tanh(\boldsymbol{W}^g \boldsymbol{x}_t + \boldsymbol{U}^g \boldsymbol{h}_{t-1}) \end{aligned} \tag{1}$$

where $\sigma(\cdot)$ and $\tanh(\cdot)$ are respectively the sigmoid function and hyperbolic tangent function. In the system of state equations (1), $\boldsymbol{W}^i$, $\boldsymbol{W}^f$, $\boldsymbol{W}^o$, and $\boldsymbol{W}^g$, as well as $\boldsymbol{U}^i$, $\boldsymbol{U}^f$, $\boldsymbol{U}^o$, and $\boldsymbol{U}^g$, are the underlying parameter matrices that determine the LSTM cell signals ($\boldsymbol{i}_t$, $\boldsymbol{f}_t$, $\boldsymbol{o}_t$), and the memory $\widetilde{\boldsymbol{m}}_t$, given the current input vector $\boldsymbol{x}_t$ and hidden layer output vector $\boldsymbol{h}_{t-1}$ at the previous time step. All parameter matrices in the LSTM cell are to be determined from the training process with a given training dataset.

Given the memory $\widetilde{\boldsymbol{m}}_t$ computed in (1) and memory $\boldsymbol{m}_{t-1}$ obtained at previous time $t-1$, the memory variable $\boldsymbol{m}_t$ at time t is updated as:

$$\boldsymbol{m}_t = \boldsymbol{i}_t \odot \widetilde{\boldsymbol{m}}_t + \boldsymbol{f}_t \odot \boldsymbol{m}_{t-1} \tag{2}$$

where the notation '$\odot$' in (2) refers to the Hadamard product, or element-wise product (Goodfellow et al., 2016; Murphy, 2022), of the two matrices. The layer output vector $\boldsymbol{h}_t$ of the LSTM cell at time t is determined as:

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{m}_t) \tag{3}$$

with which the eventual predicted output feature vector $\boldsymbol{h}_t$.

In the TFES, the LSTM cells are connected and proceed over time steps $t-1$ via the corresponding output vector of each LSTM cell. Eventually, an output vector $\boldsymbol{h}_{t-1}$, with a length of N_{LSTM}, is to be delivered with tunable parameters by the TFES based on the observations made in previous time steps, say ($\boldsymbol{x}_{t-1}$, $\boldsymbol{x}_{t-2}$, ...,$\boldsymbol{x}_{t-5}$), shown in Figure 1. The output feature of the LSTM module will be concatenated with that produced by the CNN module (see the following section) to form a comprehensive spatio-temporal feature vector for traffic observations that would lead to the eventual predicted interval of journey times through the proposed deep learning framework (see Figure 1).

2.2. Spatio-temporal feature extraction stream (STFES)

Associated with the TFES presented above is the STFES built by a CNN shown in Figure 1 for handling multi-dimensional data (Krizhevsky et al., 2012; Wu et al., 2018). Each data matrix $\boldsymbol{X}_{(j)}^{ST} = [\boldsymbol{X}_{(j)}^{ST}(t,k)]$ in the input tensor $\mathcal{X}^{ST}$ to the CNN module, where $j = 1, 2, ..., J$, contains traffic data collected over times t (where $t = 1, 2, ..., T$) and from all selected locations k (where $k = 1, 2, ..., K$) on each specific day j. The input tensor $\mathcal{X}^{ST}$ will be fed through three connected convolutional layers which generate the output of sizes ($K \times T \times N_1$), ($K \times T \times N_2$), and ($K \times T \times N_3$) respectively. The convolutional layers consist of a set of filters, known as 'kernels', with trainable parameters. During each forward pass across the convolutional layers, each kernel is convoluted across the width and height of the input tensor, which computes the dot product between the filter entries and the input and hence produces an activation map

of that input. The dimension of the feature tensor outputted from the convolutional layers is to be reduced from $(K \times T \times N_3)$ to $(Q \times Q \times N_3)$, where $Q < K$ and $Q < T$, by passing it through a pooling layer (Goodfellow et al., 2016). The principle of the pooling layer is to reduce the dimensions of the data by combining the outputs of neuron clusters at the previous convolutional layer into a single neuron in the pooling layer. This study adopts a max pooling approach which takes the maximum value of each local cluster of neurons as the representative value of that cluster during the pooling process (Goodfellow et al., 2016).

The STFES (i.e. the CNN module) will eventually generate an output feature vector with a length of N_{CNN} (see Figure 1) after passing through the max-pooling layer to the fully connected layer. This output feature will be concatenated with that produced by the TFES (i.e. the LSTM module) (see the previous section) to form a complete extracted feature vector, with a total length of $(N_{LSTM} + N_{CNN})$, from the heterogeneous traffic data (i.e. $\boldsymbol{X}_t$). The concatenation of the extracted features from the two streams, i.e, the comprehensive spatio-temporal feature vector, will be passed forward to a series of connected hidden layers, which eventually generate the corresponding upper bound $\widehat{U}(z_t)$ and lower bound $\widehat{L}(z_t)$ of the future vehicle journey time z_t at time t (see Figure 1).

2.3. Performance metrics

The objective of the prediction framework is to derive an upper bound $\widehat{U}(z_t)$ and a lower bound $\widehat{L}(z_t)$ for the future vehicle journey time z_t of a driver embarking at time t, such that the probability of this unknown future journey time z_t lying within the interval $[\widehat{L}(z_t), \widehat{U}(z_t)]$ would be greater than or equal to a predefined threshold $(1-\alpha)$, where $\alpha \in [0, 1]$, i.e.

$$\mathbb{P}[\widehat{L}(z_t) \leq z_t \leq \widehat{U}(z_t)] \geq 1 - \alpha. \tag{4}$$

The notation α in (4) can be regarded as analogous to the notion of significance level in statistical analysis. Considering the criterion specified in (4), we first define an empirical metric known as that prediction interval coverage probability ($PICP$) which measures the reliability of the estimates.

Suppose there is a total of I realized values of the vehicle journey times over the study period T. We define $z^*_{t(i)}$ be the $i-$th observed value in the dataset of I taken from a driver embarking at time t. We further define

$$c_i = \begin{cases} 1, & \text{if } z^*_{t(i)} \in [\widehat{L}(z_t), \widehat{U}(z_t)] \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

as a binary indicator denoting whether this specific observed value $z^*_{t(i)}$ fell within the prediction interval $[\widehat{L}(z_t), \widehat{U}(z_t)]$ generated by the prediction framework.

The $PICP$ can then be determined as

$$PICP = \frac{|\mathbf{c}|}{I} \tag{6}$$

where $PICP \in [0, 1]$ and $|\mathbf{c}| = \sum_{i=1}^{I} c_i$ is the total number of c_i in the set $\mathbf{c} = \{c_1, c_2, ..., c_I\}$ equaling to one. A satisfactorily trained prediction algorithm shall deliver

a $PICP \geq (1-\alpha)$ according to the criterion specified in (4) with consideration of all noises and uncertainties existing in the training dataset and prevailing traffic conditions.

In addition to the $PICP$ measure (which represents the reliability of the estimates), we have another metric for evaluating its precision known as the mean prediction interval width ($MPIW$) (Pearce et al., 2018). This $MPIW$ indicator is defined as

$$MPIW = \frac{1}{T}\sum_{t=1}^{T}[\widehat{U}(z_t) - \widehat{L}(z_t)], \tag{7}$$

which is essentially the time average of the prediction interval widths $[\widehat{L}(z_t), \widehat{U}(z_t)]$ over all times t in the study period T.

The $PICP$ and $MPIW$ in (6) and (7) will both be used for evaluating the prediction framework during the validation and testing processes. Given a target $PICP$ to achieve, a prediction associated with a smaller $MPIW$ would imply it is more precise (and hence more desirable) than another one with a larger $MPIW$.

2.4. Training procedure

We now present the training procedure for the proposed prediction. We define θ be the set of all parameters (including all parameters in the LSTM and CNN modules, and all fully connected layers) specifying the prediction framework. The training process primarily aims to determine a set of these model parameters θ that can minimize the $MPIW$ while ensuring the predefined minimum PICP can also be achieved.

To incorporate the $MPIW$ in the training process with the c_i determined in (5), we first revise the $MPIW$ indicator in (7) and derive the following loss function representing the precision of the prediction model in the training process (Pearce et al., 2018):

$$\mathcal{L}_{MPIW} = \frac{1}{I}\sum_{i=1}^{I}[\widehat{U}(z_t) - \widehat{L}(z_t)]c_i. \tag{8}$$

The loss function $\mathcal{L}_{MPIW}$ in (8) is a measure of $MPIW$ in (7) based on the realized c_i. This loss function can be regarded as the average of all prediction interval width of all data points fell within the associated interval $[\widehat{L}(z_t), \widehat{U}(z_t)]$ (i.e. c_i) over time t in the study period. The objective of this construction is to reduce the influence of outliers, i.e. data points lying outside the prediction interval, when evaluating $MPIW$ in the training process.

We now establish the loss function $\mathcal{L}_\theta$ for determining θ aiming to achieve the target $PICP$ with consideration of computational effectiveness in the training process.

Proposition 2.1. *Given a predefined target $PICP \geq (1-\alpha)$, the prediction framework parameters θ can be determined by minimizing the following loss function:*

$$\mathcal{L}_\theta = \mathcal{L}_{MPIW} + \lambda\frac{I}{\alpha(1-\alpha)}max((1-\alpha) - PICP, 0)^2 \tag{9}$$

where λ is a penalty cost balancing the trade-off between $PICP$ and $MPIW$ in the training process.

Proof. It is noted that each c_i in (5) can be regarded as following a Bernoulli process, i.e.

$$c_i \sim Bernoulli(1-\alpha) \tag{10}$$

which are identically and independently distributed for all $i = 1, 2, ..., I$ given $(1-\alpha)$ is the predefined minimum $PICP$ to achieve. The Bernoulli distributed c_i implies the corresponding total number of c_i, i.e. $|\mathbf{c}| = \sum_{i=1}^{I} c_i$, in the training dataset would follow a binomial distribution, i.e.

$$|\mathbf{c}| \sim Binomial(I, (1-\alpha)) \tag{11}$$

where I is that total observed data points and the probability of finding an observation c_i in $\mathbf{c}$ is $(1-\alpha)$. The likelihood expression of having $|\mathbf{c}|$ observations within the prediction interval can hence be specified through the Binomial probability mass function as

$$P(|\mathbf{c}||\theta) = \binom{I}{|\mathbf{c}|}(1-\alpha)^{|\mathbf{c}|}\alpha^{(I-|\mathbf{c}|)}. \tag{12}$$

It is noted that $P(|\mathbf{c}||\theta)$ in (12) is dependent on the choice of θ.

Seeking a set of model parameters θ that maximizes $P(|\mathbf{c}||\theta)$ involves computation of the combination in (12) which could be computationally demanding with a large dataset. Following Pearce et al. (2018), this study adopts a solution approach with use of Gaussian approximation on the likelihood function in (12). It is known from the central limit theorem that the binomial distribution in (12) can be approximated as a normal distribution:

$$\mathcal{N}(I(1-\alpha), I\alpha(1-\alpha)) \tag{13}$$

with the mean and standard deviation being $I(1-\alpha)$ and $I\alpha(1-\alpha)$ respectively when the underlying sample size I is sufficiently large (say, $I \geq 50$). The likelihood function in (12) can now be transformed with this Gaussian approximation as

$$P(|\mathbf{c}||\theta) = \frac{1}{\sqrt{2\pi I\alpha(1-\alpha)}} exp\Big(\frac{-(|\mathbf{c}| - I(1-\alpha))^2}{2I\alpha(1-\alpha)}\Big) \tag{14}$$

in which $|\mathbf{c}|$ is influenced by the choice of θ.

Maximizing (14) is equivalent to minimizing the following:

$$\frac{(I(1-\alpha) - |\mathbf{c}|)^2}{I\alpha(1-\alpha)}. \tag{15}$$

Substituting (6) into (15), Expression (15) can be rewritten in terms of $PICP$ as

$$\frac{I}{\alpha(1-\alpha)}((1-\alpha) - PICP)^2 \tag{16}$$

Expression (16) can be adopted as a constraint governing the choice of θ with consideration of the required minimum $PICP$ needed. We further consider that penalty should only be imposed when the resultant $PICP$ is less than the required $(1-\alpha)$ but not higher. Consequently, Expression (16) can be further revised as Pearce et al. (2018):

$$\frac{I}{\alpha(1-\alpha)} max((1-\alpha) - PICP, 0)^2 \tag{17}$$

Expression (17) can then be augmented with the $MPIW$ objective in (8) to form the Lagrangian function shown in (9), in which λ can be regarded as a penalty factor, or Lagrange multiplier, ensuring the required minimum PICP is satisfied while minimizing the $MPIW$.

□

The minimization of (9) involves computation of $|\mathbf{c}|$ for each candidate parameter set via (5). Due to the non-differentiable nature of each c_i in (5), the resultant minimization would be a combinatorial problem which needs to be solved by probabilistic search heuristics such as the particle swarm optimization (PSO) method (Khosravi et al., 2011). Such solution strategy could be computationally extensive when a large amount of data and parameter sets are involved. To further facilitate the solution process, the non-differentiable c_i in (5) is approximated by using the sigmoid function as (Khosravi et al., 2011; Murphy, 2022):

$$\tilde{c}_i = \Big[\frac{\exp[s(z^*_{t(i)} - \hat{L}(z_t))]}{\exp[s(z^*_{t(i)} - \hat{L}(z_t))] + 1}\Big]\Big[\frac{\exp[s(\hat{U}(z_t) - z^*_{t(i)})]}{\exp[s(\hat{U}(z_t) - z^*_{t(i)})] + 1}\Big] \tag{18}$$

for all observed $z^*_{t(i)}$ with the associated $\hat{L}(z_t)$ and $\hat{U}(z_t)$, governed a user-defined scaling factor s. With the approximation (18) on c_i, the $PICP$ term in the Lagrangian function (9) would become continuous with respect to the choice of parameters θ that can be minimized by effective solution algorithm such as the Adam stochastic gradient search method (Kingma and Ba, 2014). The Adam stochastic gradient search is to be used in this study for training the prediction framework in the case study presented in the following section.

3. Case Study

3.1. Scenario settings

A case study using traffic data collected from a 9-km Hong Kong strategic route section is conducted to evaluate the effectiveness of the proposed framework for interval prediction of vehicle journey times. The selected route section is shown in Figure 3 which starts from the Island Eastern Corridor (marked by the 'Entry' hexagon in the figure) and ends at the exit of Western Harbour Crossing in Kowloon (marked by the 'Exit' hexagon in the figure). The route section takes approximately 500 seconds to traverse under free-flow conditions.

The data were collected and processed over a 15-month period of weekdays from January 2017 to March 2018, 06:30 to 21:00. Traffic data of journey times, local speeds, and weather are adopted herein. To measure the journey times, a pair of Automatic

Figure 3. Case study corridor - Hong Kong strategic route

Vehicle Identification (AVI) stations have been deployed at the 'Entry' and 'Exit' locations (represented by the two hexagons in the figure) from which the journey times of the vehicles passed by can be derived from matching their identifications with the associated time stamps (Cheng et al., 2014; Chow et al., 2014; Robinson and Polak, 2006). Associated with the AVI stations are five Autoscope detection stations deployed at different locations along the route. The Autoscope stations equipped with point detectors are used to measure and record local spot speeds of vehicles through Autoscope video detection and analysis. The collected spot speed data are aggregated by the detection system into 5-min averages. We further have precipitation data, in terms of rainfall intensity per minute, in the area obtained from the Hong Kong Observatory for investigating the relationship between traffic and weather conditions.

It is known that journey times inferred from AVI could be subject to various disturbances and outliers caused by mismatching, vehicles making stop(s) or taking a detour between the AVI stations (Dion and Rakha, 2006; Robinson and Polak, 2006; Tam and Lam, 2011). To remove the invalid measurements, we adopt a filtering algorithm developed from the Spatio-Temporal Density-Based Spatial Clustering of Applications with Noise method (ST-DBSCAN) (Behara et al., 2021; Birant and Kut, 2007; Ester et al., 1996; Yao and Qian, 2021). An example of the filtered journey time data compared with its raw input is shown in Figure 4.

The dataset is divided into three subsets: data collected during 1 January 2017 - 31 October 2017 are classified as 'training dataset'. The training dataset is for determining the model parameters given a predefined model architecture. Data collected during 15 November 2017 - 20 December 2017 are used as 'validation dataset', which is for determining hyperparameters of the modeling framework that reflect the structural settings and architecture of the prediction model. The hyperparameters in the proposed prediction framework include the penalty cost λ in Equation (15), the sizes of the three convolutional layers in the CNN module (N_1, N_2, and N_3), the size of the max pooling layers (Q), and the number of units in the hidden layer in the LSTM module. Data collected during 1 March 2018 and 31 March 2018 are used as 'testing dataset' for evaluating the performance of the trained and validated modeling framework.

We now define Δt to be the sampling interval of the traffic data, which is set to 5-min as aforementioned. Considering the data granularity and practical usage in Hong

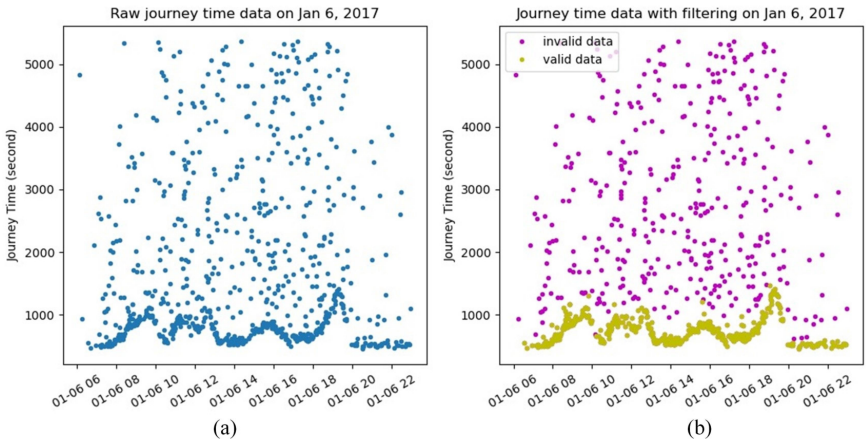


Figure 4. Vehicle journey times inferred from AVI on 6 January 2017: (a) raw data; (b) filtered data with ST-DBSCAN.

Kong, the prediction framework is set to generate an interval estimate for journey time along the selected route after 5 minutes (i.e. Δt) based on all observations over the past 25 minutes (i.e. $5\Delta t$). Accordingly, the number of cells in the LSTM module is set to be 5 (i.e. the number of sampling intervals taken from observed previous journey times) for incorporating the traffic features in the previous 25-minute period (or $5\Delta t$). The dimension of the input tensor $\mathcal{X}^{ST}$ to the CNN module, $K \times T \times J$, hence is set as $K = 5$ which refers to the number of local speed stations (corresponding to the '5' Autoscope stations) taken for generating the journey time predictions, $T = 5$ which is the number of sampling interval taken from the past data for generating the journey time predictions, the last 'J' will be subject to the number of previous days we took from the historical dataset for generating journey time predictions. The effect of setting different tensor structures and the corresponding value of J will be investigated and presented in latter parts of the paper.

The prediction framework is learned through the Adam stochastic gradient algorithm (Goodfellow et al., 2016; Kingma and Ba, 2014; Ying et al., 2020) with a maximum number of epochs of training set to be 20, the underlying learning rate in the Adam algorithm is set to be 0.0001 following the numerical experiments conducted in Goodfellow et al. (2016); Kingma and Ba (2014). The computations are coded in Python and are conducted on a computer with an Intel Core i7-10700 processor with 32GB RAM running on Windows 10.

3.2. Training and validation with different framework settings

Table 1 shows four different model set-ups representing different degrees of day-to-day correlations incorporated. The first (P0) is the base case setting in which we do not take into account of day-to-day correlation in journey times. This gives the dimension J in the input tensor $\mathcal{X}^{ST}$ to the CNN module being 'one' as we only consider observations on the present day. The second (P1) model type captures the correlation between days a week apart, say journey times observed on 'Wednesday' in the current week compared with those observed on the 'Wednesday' in the previous week. The underlying consideration here is the similarity in travel patterns on the same day in a week. The third (P2) mode type captures the correlation between consecutive days, which can be understood as the conventional day-to-day dynamics investigated

Table 1. Summary of model set-ups with different degrees of day-to-day correlations incorporated

Model type	Description
P0	No day-to-day correlation considered
P1	Same days in previous week(s)
P2	Previous day(s)
P3	Combination of P1 and P2 above

in the literature (Balijepalli et al., 2007; Zhong et al., 2020). The last model type (P3) is constructed as a combination of P1 and P2, i.e. with consideration of correlations over both consecutive days and same days over consecutive weeks.

Following a series of validation tests with data collected on 15 November 2017 - 20 December 2017 as aforementioned, the channel sizes of the three convolutional layers (N_1, N_2, and N_3) in the CNN module are set respectively to be 64, 128, and 256. Thus, the sizes of such three convolutional layers are $5 \times 5 \times 64$, $5 \times 5 \times 128$, and $5 \times 5 \times 256$ respectively based on the convolutional operation with use of 3×3 convolutional kernels. The size of the associated max pooling layers is set to be $2 \times 2 \times 256$ (i.e. $Q = 2$). The dimension of the features extracted from the CNN module (i.e. N_{CNN}) is set to be 128, while the dimension of features extracted from the LSTM module (i.e. N_{LSTM}) is set to be 64. This gives the dimension of the concatenated features from the CNN and LSTM modules $N_f = 128 + 64 = 192$. The concatenated $N_f = 192$ features are further passed through two fully connected layers with the size of 64×1 and 2×1, which lead eventually to the upper and lower bounds of the estimated journey time. The scalar s in the sigmoid function (18) is set to be 50 following our validation experiments and the analysis reported in Pearce et al. (2018).

Table 2 shows the validation results of the proposed prediction framework with different settings given their best sets of parameters and hyperparameters. Following the usual practice (Khosravi et al., 2011), the target $PICP$ (i.e. $(1-\alpha)$) is set to be 90% in the training process. The performance of the prediction framework is measured in terms of the corresponding $MPIW$ achieved. The second column in the table refers to the structure of the input tensor to the CNN module in the prediction framework. The tensor structure represents the degree of day-to-day correlations to be considered when deriving the journey time predictions. In the case of P1 where correlations between journey times observed on the same weekday over consecutive weeks are considered, the tensor structure $\mathcal{X}^{ST}_{j,j-7}$ refers to the model setting that would incorporate journey times observed on the present day j and those on the same weekday in the previous week (i.e. $j-7$). Consequently, this gives the dimension $J = 2$ with two days (i.e. j and $j-7$) of observed journey time variations considered. The tensor structure $\mathcal{X}^{ST}_{j,j-7,j-14}$ refers to the model setting that would incorporate journey times observed one week (i.e. $j-7$) and two weeks (i.e. $j-14$) before, and so forth. Consequently this specific tensor setting gives the dimension $J = 3$ with three days (i.e. j, $j-7$, and $j-14$) of observed journey time variations considered. For the case of P2, the tensor structure $\mathcal{X}^{ST}_{j,j-1}$ refers to the model setting that would incorporate correlations in journey times observed on the present day j and those on the previous day (i.e. $j-1$). Consequently, this again would give the dimension $J = 2$ with two days (i.e. j and $j-1$) of observed journey time variations considered. The tensor $\mathcal{X}^{ST}_{j,j-1,j-2}$ refers to the model setting that would incorporate correlations in journey times observed over the previous two days (i.e. $j-1$ and $j-2$), and so forth. The tensor structure settings in the table hence represent different orders of day-to-day correlations incorporated in the prediction framework.

In Table 2, the tensor structures in bold represent the best settings which we recognize for each specific model type (P1, P2, and P3). We regard a model setting is 'better' than another if it produces a smaller $MPIW$ given that the minimum required $PICP$ is achieved (i.e. 90%). We note the tensor settings $\mathcal{X}^{ST}_{j-7,j-14}$ (P1), $\mathcal{X}^{ST}_{j-1,j-2}$ (P2), and $\mathcal{X}^{ST}_{j-1,j-2,j-7,j-14}$ (P3) perform the best in their own type with consideration of their $PICP$ and $MPIW$ loss function values achieved.

For further reference, Table 2 also includes the point estimation metrics: mean absolute percentage error ($MAPE$), mean absolute error (MAE), and root mean square error ($RMSE$). Given the estimated interval $[\hat{L}(z_t), \hat{U}(z_t)]$ derived for each journey time z_t over time t with which $z^*_{t(i)}$ is the associated true ($i-$th) observed value, we define

$$\hat{z}_t = \frac{\hat{U}(z_t) + \hat{L}(z_t)}{2} \tag{19}$$

be the point estimate for z_t derived from the interval $[\hat{L}(z_t), \hat{U}(z_t)]$, assuming the distribution of true value of z_t within the interval is uniform. We further have the corresponding point estimation metrics over all observed data points as:

$$MAE = \frac{\sum_{i=1}^{I} |\hat{z}_t - z^*_{t(i)}|}{I} \tag{20}$$

$$MAPE = \frac{\sum_{i=1}^{I} |\hat{z}_t - z^*_{t(i)}|/z^*_{t(i)}}{I} \tag{21}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{I} |\hat{z}_t - z^*_{t(i)}|^2}{I}} \tag{22}$$

Table 2 reveals that all settings herein can achieve the minimum predefined target $PICP$ (90%) in the validation test. It also indicates that the settings under P3 (i.e. with consideration of both weekly and daily day-to-day journey time correlations) deliver the best performances in terms of both $PICP$ and $MPIW$, as well as all the point estimation metrics ($MAPE$, MAE, $RMSE$) achieved when compared with other settings. The results generally show that incorporation of the day-to-day correlations could improve the precision and reliability of the vehicle journey time prediction.

Figure 5 further illustrates the progression of the training processes in terms of the reduction of the loss function values over iterations (epochs) with the best tensor structures identified in Table 2 for each model type. During the training process, the parameters in the LSTM and CNN components are refined iteratively through the Adam stochastic gradient algorithm. For validation purpose, the trained model by each epoch is also used to generate predictions using data in the validation dataset. The validation results are included in the figure as shown. The validation process provides an unbiased evaluation of the trained models with a dataset that is different from the one used for training.

Figure 5 shows that the most significant improvements would occur in the first five epochs, followed by more gradual improvement until Epoch 20 where there is no fur-

Table 2. Performances of different prediction settings in the validation process

Type	CNN tensor structure	Prediction Interval		Point Estimation		
		$PICP$	$MPIW$(s)	$MAPE$	MAE(s)	$RMSE$(s)
P0	$\mathcal{X}_{j}^{ST}$	**91.6**%	**309.8**	**7.7**%	**71.1**	**111.8**
P1	$\mathcal{X}_{j,j-7}^{ST}$	93.1%	327.7	7.9%	71.8	112.4
	$\mathcal{X}_{j,j-7,j-14}^{ST}$	**93.3**%	**320.0**	**7.8**%	**69.6**	**109.5**
	$\mathcal{X}_{j,j-7,j-14,j-21}^{ST}$	92.0%	331.4	8.0%	74.2	116.4
P2	$\mathcal{X}_{j,j-1}^{ST}$	93.6%	343.7	8.5%	74.8	112.2
	$\mathcal{X}_{j,j-1,j-2}^{ST}$	92.5%	321.4	7.8%	71.6	112.1
	$\mathcal{X}_{j,j-1,j-2,j-3}^{ST}$	**91.9**%	**319.1**	**7.7**%	**72.6**	**115.9**
P3	$\mathcal{X}_{j,j-1,j-7}^{ST}$	92.3%	322.1	7.6%	70.3	112.1
	$\mathcal{X}_{j,j-1,j-2,j-7,j-14}^{ST}$	**92.3**%	**300.5**	**7.4**%	**68.2**	**109.0**
	$\mathcal{X}_{j,j-1,j-2,j-3,j-7,j-14,j-21}^{ST}$	90.5%	345.0	8.1%	77.9	125.7

ther noticeable improvement in the loss function values. The minimum requirement on the $PICP$, i.e. $(1-\alpha)$, is satisfied in all training and validation processes presented herein. Consequently, the loss function values reflect the precision, i.e. $MPIW$, that each model setting could achieve with the minimum $PICP$ satisfied. Despite the differences in the performances (i.e. loss function values) achieved, the validation processes essentially follow the same trend as the training processes. This implies an improvement achieved by the model parameter adjustments in each training epoch would also result in a corresponding improvement in the validation epoch. This can be interpreted as the fact that overfitting does not occur in the training process in the current settings (Yang et al., 2019). As expected, the trained prediction framework delivers better performance, in terms of a lower loss function value, when a higher degree of day-to-day correlation is incorporated.

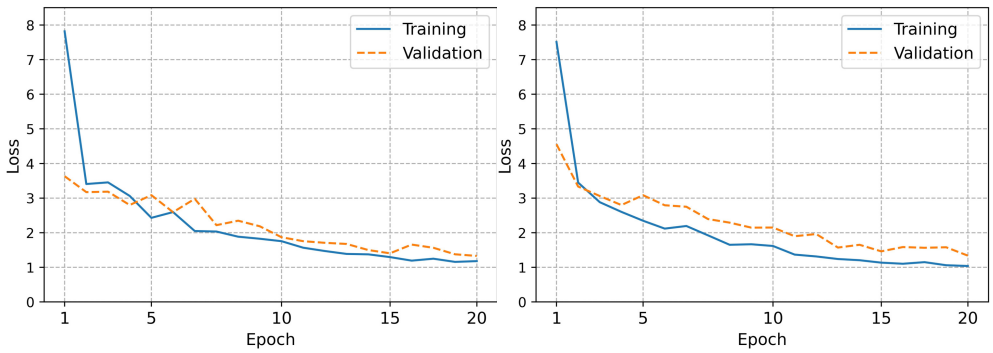
3.3. Determination of hyperparameter λ

As an illustration on determining the hyperparameter λ through the validation process, Figure 6 displays the loss function values $\mathcal{L}_{PICP}$ and $\mathcal{L}_{MPIW}$ achieved in the validation processes over different λ. In the figure, we adopt the 'P3' model setting with CNN input tensor structure $\mathcal{X}_{j,j-1,j-2,j-7,j-14}^{ST}$ as an illustration. The loss function $\mathcal{L}_{PICP}$ here is defined as

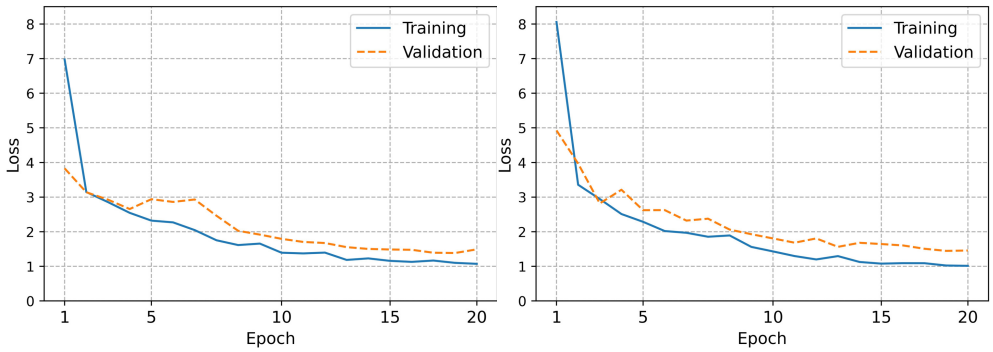
$$\mathcal{L}_{PICP} = \frac{I}{\alpha(1-\alpha)} max((1-\alpha) - PICP, 0)^2 \tag{23}$$

as extracted from (9) in which the $PICP$ is calculated with use of approximation on c_i in (18). The loss function values $\mathcal{L}_{PICP}$ and $\mathcal{L}_{MPIW}$ in Figure 6 are calculated with normalized data instead of actual data as discussed in Section 2.

As reflected in (9), a larger λ implies more penalty would be placed on $PICP$ which leads to the increase of $\mathcal{L}_{MPIW}$ (i.e. reduction in precision) in order to cover more data points within the prediction interval and hence result in a lower $\mathcal{L}_{PICP}$ (i.e. improvement in reliability). As observed from the figure, there is no significant reduction in $\mathcal{L}_{PICP}$ for $\lambda > 0.5$. Consequently, we decide from this validation process that the value of λ shall be set to be 0.5 for this case as a balanced consideration of both $PICP$ and $MPIW$ to be achieved.



(a) P0 (b) P1



(c) P2 (d) P3

Figure 5. Training and validation processes for different prediction framework settings

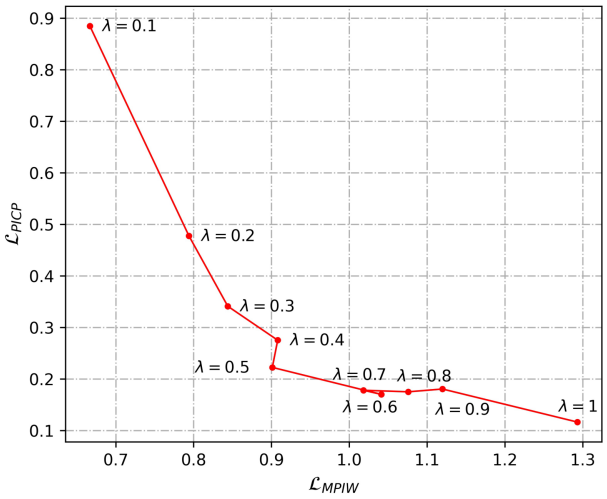


Figure 6. Example of loss function values achieved with different λ in the validation process for 'P3' model with CNN input tensor structure $\mathcal{X}^{ST}_{j,j-1,j-2,j-7,j-14}$

Table 3. Performances of the best model settings for each type in the testing process

Type	CNN tensor structure	Prediction Interval		Point Estimation		
		$PICP$	$MPIW$(s)	$MAPE$	MAE(s)	$RMSE$(s)
P0	$\mathcal{X}_j^{ST}$	95.0%	311.3	6.5%	57.8	86.5
P1	$\mathcal{X}_{j,j-7,j-14}^{ST}$	95.3%	330.3	7.0%	61.2	90.9
P2	$\mathcal{X}_{j,j-1,j-2,j-3}^{ST}$	93.9%	317.9	7.1%	66.2	99.7
P3	$\mathcal{X}_{j,j-1,j-2,j-7,j-14}^{ST}$	93.8%	305.3	6.9%	62.7	94.2

3.4. Testing results

The best four CNN input tensor settings identified from the training and validation results (i.e. $\mathcal{X}_j^{ST}$ (P0), $\mathcal{X}_{j,j-7,j-14}^{ST}$ (P1), $\mathcal{X}_{j,j-1,j-2,j-3}^{ST}$ (P2), and $\mathcal{X}_{j,j-1,j-2,j-7,j-14}^{ST}$ (P3)) are now evaluated through the testing process. The testing results are summarized in Table 3. The results are generally consistent with the training and validation ones, and they reveal that the proposed prediction framework can incorporate day-to-day correlation and deliver the best performance in terms of $MPIW$ under the P3 setting. Nevertheless, the advantage of incorporating the day-to-day journey time correlation is not as significant as that reflected in the training and validation stages, as the testing results reveal that the basic P0 model setting could still deliver a similar or even better performance in some occasions when compared with the other more sophisticated settings (P1 and P2). This suggests that the within-day journey time variations play a more important role in the prediction framework than the day-to-day variations and correlations. This echoes some of our previous findings (Cheng et al., 2014). It can be understood for the fact that the within-day information would capture more relevant features such as prevailing traffic variations due to roadwork, incidents, and adverse weather. These more recent features contribute more significantly to the current traffic conditions compared with the traffic features recorded on previous days or weeks.

3.5. Benchmarking with existing methods

The performance of the proposed prediction framework is compared with five established algorithms in the literature for benchmarking purposes. The five algorithms selected here are ARIMA (AutoRegressive Integrated Moving Average) (Cheng et al., 2014), FPCA (Functional Principal Component Analysis) (Chen and Müller, 2014; Zhong et al., 2017), the conventional LSTM-based prediction algorithm (Li et al., 2021b; Yang et al., 2019), the convolution-based prediction algorithm (TCN) (Bai et al., 2018), and the convolutional LSTM neural network (ConvLSTM) (Fu et al., 2020; Petersen et al., 2019; Ran et al., 2019; Shi et al., 2015). Among the selected algorithms, ARIMA is a classical time-series model with an assumption of stationarity, which implies there are no systematic temporal variations in the time series data apart from white noises. The FPCA is a parametric time series prediction model constructed with use of functional approximation. Essentially, the FPCA model learns the relationship between the historical and target journey time series in its training procedure, and then applies the learned relationship for short-term prediction of journey times and their associated variability (Zhong et al., 2017). The LSTM predictor can be regarded as the proposed prediction framework without incorporation of the CNN module. This essentially reduces the proposed prediction framework to a pure time-series prediction process with no incorporation of multiple data sources. The TCN is another type of time-series prediction model with use of convolutional architectures.

Table 4. Overall testing results of the benchmarking algorithms

Algorithm	Prediction Interval		Point Estimation		
	$PICP$	$MPIW$(s)	$MAPE$	MAE(s)	$RMSE$(s)
ARIMA	88.1%	249.1	6.6%	59.0	91.1
FPCA	87.2%	733.5	14.8%	162.6	1045
LSTM	94.3%	350.2	7.3%	68.2	103.1
TCN	93.3%	309.5	6.9%	60.9	91.0
ConvLSTM	94.7%	345.7	6.9%	61.0	92.0

Table 5. Comparison of testing results at morning peak (7:30-10:00)

Algorithm		Prediction Interval		Point Estimation		
		$PICP$	$MPIW$(s)	$MAPE$	MAE(s)	$RMSE$(s)
ARIMA		91.8%	246.2	6.3%	52.6	75.9
FPCA		78.9%	447.5	8.5%	67.2	86.5
LSTM		95.0%	310.6	6.2%	52.1	74.5
TCN		94.5%	289.7	6.6%	55.2	77.9
ConvLSTM		94.3%	303.4	6.1%	51.4	74.3
Ours	P0	93.8%	244.8	6.0%	49.8	67.8
	P1	95.5%	283.9	6.5%	52.8	71.9
	P2	94.3%	261.5	6.4%	51.0	69.2
	P3	93.5%	273.0	6.4%	51.7	71.0

To consider the spatio-temporal feature extraction in both input-to-state and state-to-state transactions, the ConvLSTM, a single structure with a novel combination of convolutional and LSTM layers, is introduced. All benchmarking algorithms selected herein (including LSTM) operate with only a single source of data (i.e. AVI journey times). Hence the benchmarking experiments can also reveal the benefit of incorporating multiple data sources as in the proposed prediction framework. All benchmarking algorithms are trained and validated, with a target $PICP = 90\%$, using the specified training and validation datasets with their associated training processes. The overall testing results of the benchmarking algorithms are shown in Table 4 with inclusion of both interval and point estimation metrics.

Compared with the results given in Table 3, it is found that the proposed approach could outperform the benchmarking algorithms with a lower $MPIW$ given the same $PICP$ target. The proposed prediction framework can also deliver lower point estimation metrics: $MAPE$, MAE, and $RMSE$ when compared with the benchmarking algorithms. As a stationary time-series model (Cheng et al., 2014), ARIMA is not effective in capturing the transient characteristics of road traffic for dynamic modeling purpose and it is computational expensive for one-step prediction. Moreover, it

Table 6. Comparison of testing results at afternoon peak (17:30-19:30)

Algorithm		Prediction Interval		Point Estimation		
		$PICP$	$MPIW$(s)	$MAPE$	MAE(s)	$RMSE$(s)
ARIMA		81.2%	251.9	6.9%	79.4	111.6
FPCA		91.7%	1014.6	10.2%	120.3	167.7
LSTM		93.2%	493.8	8.1%	101.8	141.9
TCN		93.5%	424.5	6.8%	78.1	109.8
ConvLSTM		94.5%	477.1	7.1%	84.3	116.5
Ours	P0	95.7%	448.5	6.7%	79	110.8
	P1	95.7%	461.0	6.9%	80.8	111.8
	P2	92.0%	442.0	8.0%	101.6	142
	P3	93.8%	415.4	7.2%	87.6	124.3

Table 7. Comparison of testing results in the afternoon (13:30-15:30)

Algorithm		Prediction Interval		Point Estimation		
		$PICP$	$MPIW$(s)	$MAPE$	MAE(s)	$RMSE$(s)
ARIMA		92.9%	249.4	5.6%	45.9	85.1
FPCA		85.5%	656.8	41.0%	523.6	2620.4
LSTM		96.6%	288.2	6.2%	49.8	79.9
TCN		95.4%	254.5	6.1%	48.7	84.4
ConvLSTM		97.2%	290.3	6.0%	49.5	90.3
Ours	P0	97.2%	258.4	6.0%	47.8	81.8
	P1	96.6%	276.4	6.7%	53.5	93.3
	P2	96.0%	265.5	6.4%	51.8	84.0
	P3	97.2%	259.8	6.3%	48.6	76.6

estimates the uncertainties of journety time by using deterministic predictions based on assumptions that the residuals are uncorrelated and normally distributed. Thus, alghouth ARIMA achieves a relatively good performance on point estimation, it fails to deliver a target prediction interval. Given the same target $PICP$, the proposed prediction framework could deliver a $MPIW$ that is almost halved compared with that by the parametric FPCA. We attribute this to the capability of the underlying ANN-based deep learning framework in the proposed algorithm for capturing complex traffic dynamics. This reveals and supports the deep learning method as a promising approach and research direction for complex traffic modeling compared with the conventional parametric modeling approaches. Furthermore, the convetional LSTM could deliver significantly better performance than the other two more conventional benchmarking methods. However, it is still being outperformed by around 10% in terms of the $MPIW$ delivered when compared with the proposed prediction framework which incorporates multiple data sources through the CNN module. TCN can offer similar performance on point estimation compared with the proposed framework, but it still fails to achieve a better prediction interval in terms of $MPIW$. Alghtough the ConvLSTM achives a better prediction interval compared with the convetional LSTM, which indicates the effictiveness of the convolutional layers, it is also being outperformed by around 10% in terms of the $MPIW$ compared with the proposed framework. The comparison between the proposed framework and the benchmarking algorithms highlights the benefit of incorporating information extracted from multiple data sources in terms of improving the prediction precision in a deep learning-based traffic prediction framework.

To further investigate the performance of different algorithms at different periods especially when the transportation systems are most vulnerable, morning peaks (7:30-10:00) and afternoon peaks (17:30-19:30) of weekdays in the testing dataset are considered. The comparison results of all algorithms are shown in Tables 5 and 6. The CNN tensor structures of P0, P1, P2, and P3 of the proposed framework are the same as that described in Table 3. The proposed framework outperforms all benchmarks on prediction intervals for morning peak hours and afternoon peak hours. It supports that the proposed framework is able to capture dynamic traffic states and deliver a better uncertainty estimation for irregular travel time. Compared with rush hours, a relatively free-flow traffic period (13:30-15:30) is considered and comparison results are shown in Table 7. According to Table 7, it is found that the proposed framework can also deliver a similar performance in terms of prediction interval and point estimation compared with ARIMA. The reason why ARIMA can achieve the best performance in the free-flow traffic state is that the travel time is almost stationary and ARIMA is re-trained for each one-step prediction. The comparison of all algorithms for differ-

ent traffic periods indicates the proposed distribution-free reliability-based prediction framework has the ability to handle dynamic and complex traffic states.

3.6. Effect of weather conditions

We now investigate the effect of weather conditions incorporated into the prediction framework via the LSTM module. Two days, 14 March 2018 and 21 March 2018, have been selected from the test dataset as representatives of a rainy day and a sunny day respectively. The two days are a week apart, and hence it is reasonable to assume that the underlying travel patterns on the two days would be similar except under different weather conditions.

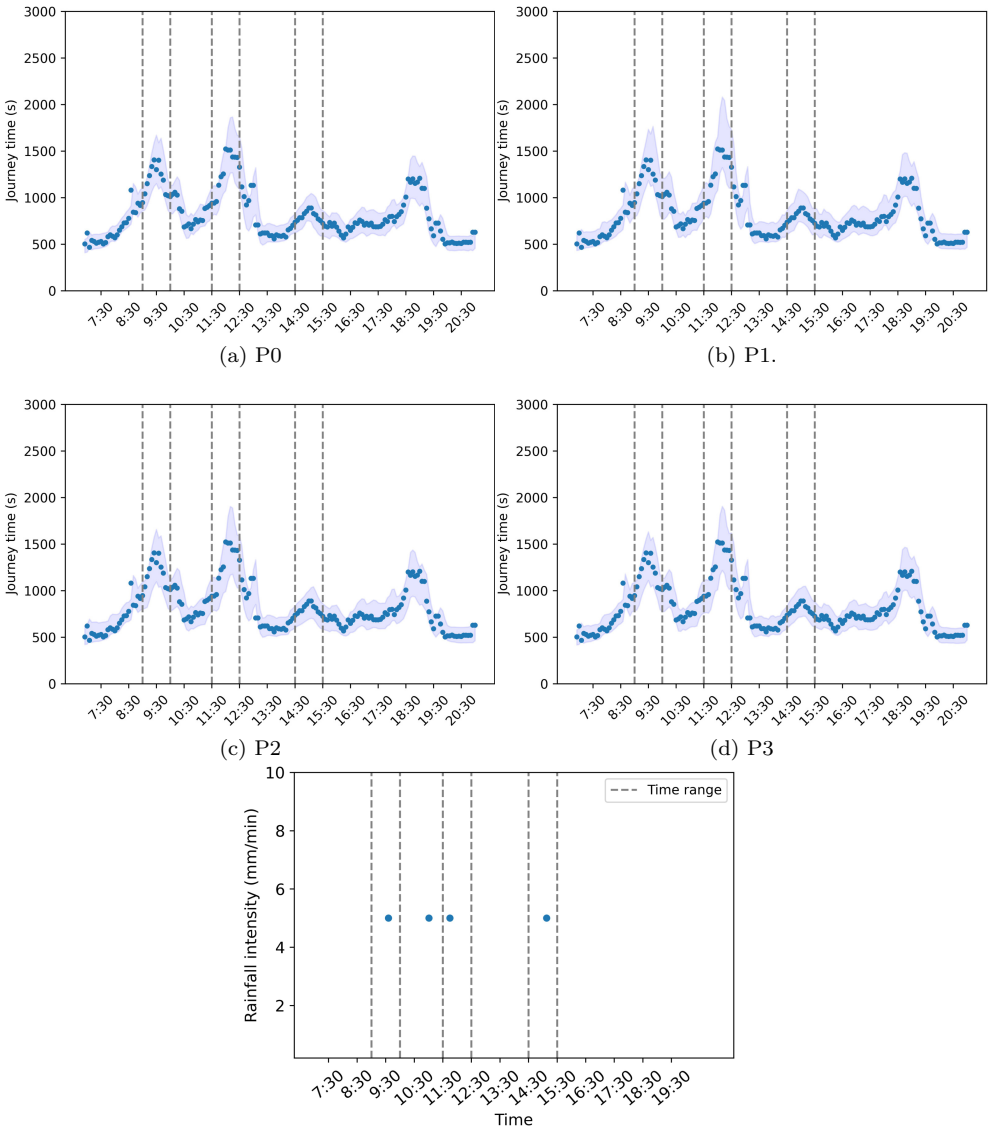
Figures 7 and 8 depict the performances of the proposed prediction framework on the two selected days over different model settings. The dots in the figures (a-d) are the actual measurements, regarded as 'ground truth', of the vehicle journey times while the shaded regions in the figures represent the corresponding journey time prediction intervals produced by the trained and validated prediction model. For readers' reference, Figure 7 (e) shows the associated precipitation record (rainfall intensity (mm/min) on 14 March 2018 provided by the Hong Kong Observatory.

On the effect of congestion, the figures first reveal that estimations with greater variability (characterized by the wider prediction intervals) are produced from the prediction algorithms during the congested hours (i.e. when the average journey times are high). This characteristic is consistent with empirical observations herein (the 'dots') as well as previous studies on journey time characteristics (Chow et al., 2014; Zhong et al., 2017).

On the effect of adverse (rainy) weather conditions, we attribute the surges in journey times during 08:30-10:30, 11:30-12:30, and 14:30-15:30 on 14 March 2018 in Figure 7 to the rainfall during those periods as compared with the journey time pattern (see Figure 8) observed on the sunny day (21 March 2018). Figure 8 reveals that the prediction algorithms are able to estimate the surges in journey times as well as the associated variability during those periods. Figur 9 shows the training and validation processes (in terms of the logarithm of the loss function values) of the prediction with and without consideration of weather conditions. Similar to Figure 5, the validation processes essentially follow the same trend as the training processes over epochs run. This implies overfitting does not occur during the training process of the prediction models with and without incorporation of weather conditions. Moreover, it is observed that the eventual training and validation processes could achieve similar loss function values for all model settings regardless of whether weather information is incorporated. This implies weather information (e.g. rainfall intensity) does not have a significant effect on improving the precision and reliability of the prediction results. We reckon this could be because the influence of the weather condition has already been reflected in the corresponding variations in traffic flows and journey times (e.g. journey times increase as rainfall intensity increases as shown in Figure 7). Consequently, the weather condition is not a crucial factor to include for improving the precision and reliability of the journey time prediction.

4. Conclusion

This paper presents a distribution-free reliability-based prediction framework for vehicle journey times and their associated variability based on a two-stream deep learning

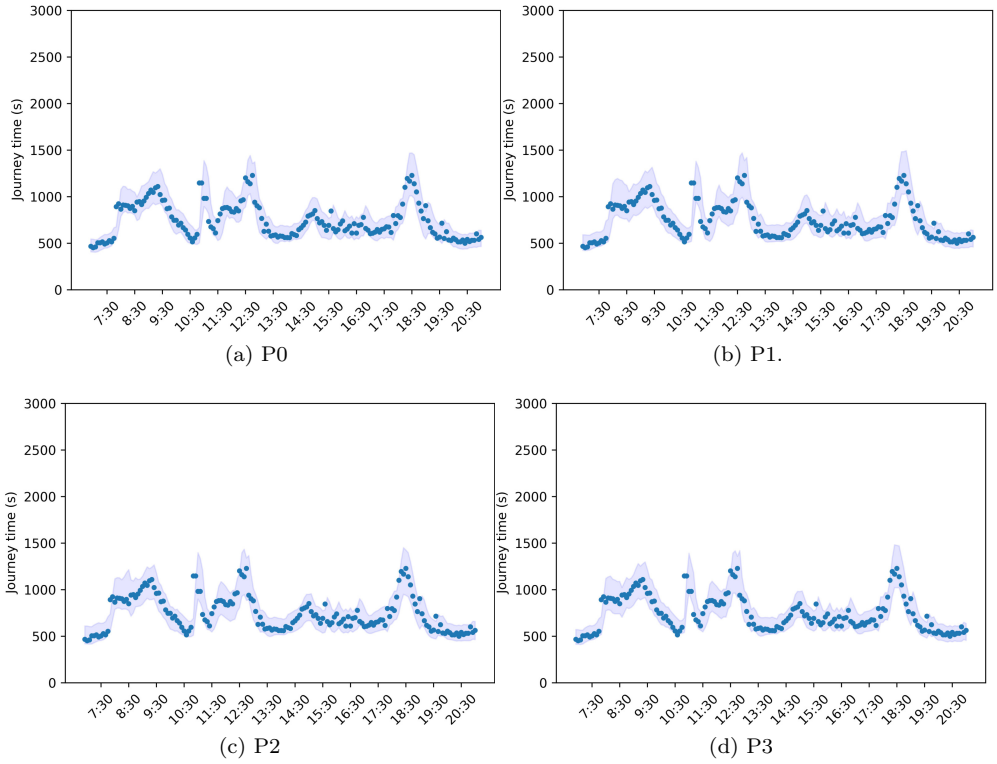


(a) P0 (b) P1.

(c) P2 (d) P3

(e) Precipitation on 14 March 2018

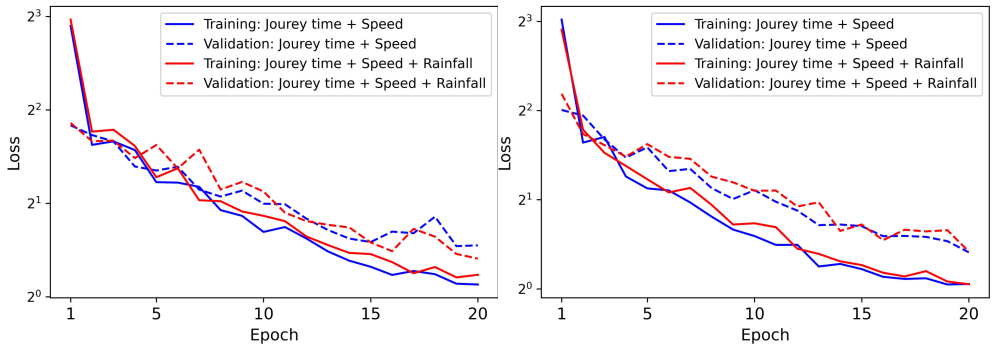
Figure 7. Prediction results on a rainy day - 14 March 2018



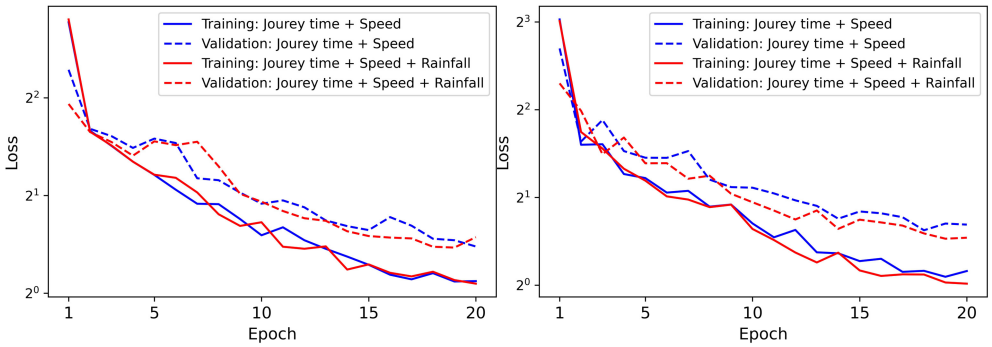
(a) P0 (b) P1. (c) P2 (d) P3

Figure 8. Prediction results on a sunny day - 21 March 2018

framework with use of heterogeneous traffic data collected from multiple sources, locations, times, and days. The heterogeneous data sources provide ranges of spatial and temporal features of prevailing traffic flow for precise and reliable journey time estimation. The spatial and temporal features in the input data are extracted through the underlying two streams (TFES and STFES) which are built respectively by LSTM and CNN. Given a target level of reliability $PICP$, the extracted features are fused and processed through a series of fully connected layers to generate estimates of lower and upper bound of an unknown vehicle journey time at a time of interest. We further introduce a smooth and differentiable loss function for the training process through use of a Gaussian approximation. The training algorithm can hence be conducted via the computational effective Adam stochastic gradient search. The proposed prediction framework is implemented and tested with traffic data collected from a selected Hong Kong corridor with sensitivity analyses conducted on different traffic and weather conditions. The proposed prediction framework is also compared with five other established journey time prediction methods (ARIMA, FPCA, LSTM, TCN, ConvLSTM) for benchmarking. The results reveal that the proposed approach can deliver estimates of journey time distributions with higher precision (i.e. lower $MPIW$) given target reliability (i.e. $PICP$). We attribute this to the capability of the underlying ANN-based deep learning framework for capturing the complex spatio-temporal traffic dynamics. The results herein support the use of deep learning methods for developing advanced traffic modeling and prediction system. The proposed framework also outperforms the conventional LSTM which is regarded as a standard deep learning-based prediction approach in the present study with a single data source. The comparison with the conventional LSTM highlights the benefit of incorporating temporal and spatial features



(a) P0 (b) P1.



(c) P2 (d) P3

Figure 9. Training and validation processes for different prediction framework settings with and without consideration of weather condition

from multiple data sources.

With different formulations of the prediction framework, it is also found that the precision and reliability of the corresponding journey time estimates could be improved through incorporation of day-to-day correlation. Nevertheless, it is observed that the value of the day-to-day journey time information is less significant than that observed on the same day for short-term journey time prediction purpose. Moreover, on the effect of including non-traffic information such as weather conditions, our test results reveal that the proposed prediction framework could deliver similar performance whether weather information is incorporated or not. This implies, despite the flexibility of the proposed prediction framework herein, inclusion of day-to-day journey time and additional (*e.g.*, weather) information may not have a significant effect on improving the precision and reliability of journey time prediction as reflected in our case study. We reckon this could be due to the fact that the influence of traffic conditions on previous day(s) and weather have already been reflected in the prevailing observations of traffic flows and journey times.

This work contributes to reliability-based journey time prediction with use of emerging deep learning techniques. It offers potential for investigating the use of advanced deep learning algorithms for development of urban traffic prediction and intelligent transportation systems with heterogeneous data sources. On building reliable intelligent transportation systems, the proposed journey time prediction algorithm could be applied to development of reliability-based travel guidance system (Chow et al., 2020) or multi-modal network management (Chow et al., 2021). Finally, it is noted that the prediction framework presented here is data-driven. Future work will also include integrating the present data-driven approach with a model-based approach (Su et al., 2021) through the use of advanced traffic flow modeling technique (Bai et al., 2021; Shi et al., 2021) for more accurate and stable results.

Acknowledgement(s)

The authors would like to thank Hong Kong Transport Department and Autotoll Limited for providing the traffic data.

Funding

This study is supported by two Research Impact Funds (RIF) sponsored by the Hong Kong Research Grants Council (Refs: R5029-18 and R7027-18).

References

Abdollahi, M., Khaleghi, T., and Yang, K. (2020). An integrated feature learning approach using deep learning for travel time prediction. *Expert Systems with Applications*, 139:112864.

Bai, L., Wong, S. C., Xu, P., Chow, A. H. F., and Lam, W. H. K. (2021). Calibration of stochastic link-based fundamental diagram with explicit consideration of speed heterogeneity. *Transportation Research Part B: Methodological*, 150:524–539.

Bai, S., Kolter, J. Z., and Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Balijepalli, C., Watling, D., and Liu, R. (2007). Doubly dynamic traffic assignment: Simulation modeling framework and experimental results. *Transportation Research Record*, 2029:39–48.

Behara, K., Bhaskar, A., and Chung, E. (2021). A DBSCAN-based framework to mine travel patterns from origin-destination matrices: Proof-of-concept on proxy static OD from Brisbane. *Transportation Research Part C*, 131:103370.

Billings, D. and Yang, J.-S. (2006). Application of the ARIMA models to urban roadway travel time prediction-a case study. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 3, pages 2529–2534. IEEE.

Birant, D. and Kut, A. (2007). ST-DBSCAN: An algorithm for clustering spatial–temporal data. *Data & knowledge engineering*, 60(1):208–221.

Chen, C.-M., Liang, C.-C., and Chu, C.-p. (2020). Long-term travel time prediction using gradient boosting. *Journal of Intelligent Transportation Systems*, 24(2).

Chen, K. and Müller, H.-G. (2014). Modeling conditional distributions for functional responses, with application to traffic monitoring via GPS-enabled mobile phones. *Technometrics*, 56(3):347–358.

Chen, Y., Chen, C., Wu, Q., Ma, J., Zhang, G., and Milton, J. (2021). Spatial-temporal traffic congestion identification and correlation extraction using floating car data. *Journal of Intelligent Transportation Systems*, 25(3):263–280.

Cheng, T., Wang, J., Haworth, J., Heydecker, B., and Chow, A. (2014). A dynamic spatial weight matrix and localized space-time autoregressive integrated moving average for network modeling. *Geographical Analysis*, 46:75–97.

Chow, A. H. F., Li, S., Szeto, W. Y., and Wang, D. (2015). Modelling urban traffic dynamics based upon the variational formulation of kinematic waves. *Transportmetrica B*, 3(3):169–191.

Chow, A. H. F., Li, S., and Zhong, R. X. (2017). Multi-objective optimal control formulations for bus service reliability with traffic signals. *Transportation Research Part B: Methodological*, 103:248–268.

Chow, A. H. F., Santacreu, A., Tsapakis, I., Tanaksaranond, G., and Cheng, T. (2014). Empirical assessment of urban traffic congestion. *Journal of Advanced Transportation*, 48(8):1000–1016.

Chow, A. H. F., Sha, R., and Li, Y. (2020). Adaptive control strategies for urban network traffic via a decentralised approach with user-optimal routing. *IEEE Transactions on Intelligent Transportation Systems*, 21(4):1697–1704.

Chow, A. H. F., Su, Z., Liang, E., and Zhong, R. X. (2021). Adaptive signal control for bus service reliability with connected vehicle technology via reinforcement learning. *Transportation Research Part C: Emerging Technologies*, 129:103264.

Connors, R. and Sumalee, A. (2009). A network equilibrium model with travellers' perception of stochastic travel times. *Transportation Research Part B: Methodological*, 43(6):614–624.

Dia, H. (2001). An object-oriented neural network approach to short-term traffic forecasting. *European Journal of Operational Research*, 131(2):253–261.

Dion, F. and Rakha, H. (2006). Estimating dynamic roadway travel times using automatic vehicle identification data for low sampling rates. *Transportation Research Part B*, 40:745–766.

Duan, Y., Yisheng, L., and Wang, F.-Y. (2016). Travel time prediction with LSTM neural network. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pages 1053–1058. IEEE.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 226–231.

Feng, X., Ling, X., Zheng, H., Chen, Z., and Xu, Y. (2018). Adaptive multi-kernel svm with spatial–temporal correlation for short-term traffic flow prediction. *IEEE Transactions on Intelligent Transportation Systems*, 20(6):2001–2013.

Fu, K., Meng, F., Ye, J., and Wang, Z. (2020). Compacteta: A fast inference system for travel time prediction. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 3337–3345.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep Learning. *MIT press, MA*.

Haworth, J., Shawe-Taylor, J., Cheng, T., and Wang, J. (2014). Local online kernel ridge regression for forecasting of urban travel times. *Transportation Research Part C: Emerging Technologies*, 46:151–178.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Khosravi, A., Nahavandi, S., Creighton, D., and Atiya, A. F. (2011). Lower upper bound estimation method for construction of neural network-based prediction intervals. *IEEE Transactions on Neural Networks*, 22(3):337–346.

Kingma, D. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25:1097–1105.

Kurzhanskiy, A. and Varaiya, P. (2010). Active traffic management on road networks: a macroscopic approach. *Philosophical Transactions of The Royal Society A: Mathematical Physical and Engineering Sciences*, 368(1928):4607–4626.

Kurzhanskiy, A. A. and Varaiya, P. (2012). Guaranteed prediction and estimation of the state of a road network. *Transportation research part C: emerging technologies*, 21(1):163–180.

Kwak, S. and Geroliminis, N. (2020). Travel time prediction for congested freeways with a dynamic linear model. *IEEE Transactions on Intelligent Transportation Systems*, 22(12):7667–7677.

Li, F., Feng, J., Yan, H., Jin, G., Yang, F., Sun, F., Jin, D., and Li, Y. (2021a). Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution. *ACM Transactions on Knowledge Discovery from Data (TKDD), In press.*

Li, J., Guo, F., Sivakumar, A., Dong, Y., and Krishnan, R. (2021b). Transferability improvement in short-term traffic prediction using stacked LSTM network. *Transportation Research Part C: Emerging Technologies*, 124:102977.

Li, L., Qu, X., Zhang, J., Wang, Y., and Ran, B. (2019). Traffic speed prediction for intelligent transportation system based on a deep feature fusion model. *Journal of Intelligent Transportation Systems*, 23(6):605–616.

Li, L., Ran, B., Zhu, J., and Du, B. (2020). Coupled application of deep learning model and quantile regression for travel time and its interval estimation using data in different dimensions. *Applied Soft Computing*, 93:106387.

Li, Z., Xu, H., Gao, X., Wang, Z., and Xu, W. (2022). Fusion attention mechanism bidirectional lstm for short-term traffic flow prediction. *Journal of Intelligent Transportation Systems*, pages 1–14.

Lin, L., Handley, J. C., Gu, Y., Zhu, L., Wen, X., and Sadek, A. W. (2018). Quantifying uncertainty in short-term traffic prediction and its application to optimal staffing plan development. *Transportation Research Part C: Emerging Technologies*, 92:323–348.

Murphy, K. P. (2022). Probabilistic machine learning: An introduction. *The MIT Press, Cambridge, MA.*

Myung, J., Kim, D.-K., Kho, S.-Y., and Park, C.-H. (2011). Travel time prediction using k nearest neighbor method with combined data from vehicle detector system and automatic toll collection system. *Transportation Research Record*, 2256(1):51–59.

Pearce, T., Brintrup, A., Zaki, M., and Neely, A. (2018). High-quality prediction intervals for deep learning: A distribution-free, ensembled approach. In *International Conference on Machine Learning*, pages 4075–4084. PMLR.

Petersen, N. C., Rodrigues, F., and Pereira, F. C. (2019). Multi-output bus travel time prediction with convolutional lstm neural network. *Expert Systems with Applications*, 120:426–435.

Qiao, W., Haghani, A., and Hamedi, M. (2013). A nonparametric model for short-term travel time prediction using bluetooth data. *Journal of Intelligent Transportation Systems*, 17(2):165–175.

Qiao, W., Haghani, A., Shao, C.-F., and Liu, J. (2016). Freeway path travel time prediction based on heterogeneous traffic data through nonparametric model. *Journal of Intelligent*

Transportation Systems, 20(5):438–448.

Ran, X., Shan, Z., Fang, Y., and Lin, C. (2019). An LSTM-based method with attention mechanism for travel time prediction. *Sensors*, 19(4):861.

Robinson, S. and Polak, J. (2006). Overtaking rule method for the cleaning of matched license plate data. *ASCE Journal of Transportation Engineering*, 132(8):609–617.

Shi, R., Mo, Z., Huang, K., Di, X., and Du, Q. (2021). A physics-informed deep learning paradigm for traffic state and fundamental diagram estimation. *IEEE Transactions on Intelligent Transportation Systems, In press*.

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., and Woo, W.-c. (2015). Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28.

Shi, X., Qi, H., Shen, Y., Wu, G., and Yin, B. (2020). A spatial–temporal attention approach for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, 22(8):4909–4918.

Su, Z., Chow, A. H. F., and Zhong, R. X. (2021). Adaptive network traffic control with an integrated model-based and data-driven approach and a decentralised solution method. *Transportation Research Part C: Emerging Technologies*, 128:103154.

Sumalee, A., Pan, T., Zhong, R. X., Uno, N., and Indra-Payoong, N. (2013). Dynamic stochastic journey time estimation and reliability analysis using stochastic cell transmission model: Algorithm and case studies. *Transportation Research Part C: Emerging Technologies*, 35:263–285.

Tam, M. L. and Lam, W. H. K. (2011). Application of automatic vehicle identification technology for real-time journey time estimation. *Information Fusion*, 12:11–19.

Tang, K., Chen, S., Khattak, A. J., and Pan, Y. (2021). Deep architecture for citywide travel time estimation incorporating contextual information. *Journal of Intelligent Transportation Systems*, 25(3):313–329.

Wang, B., Vu, H. L., Kim, I., and Cai, C. (2022). Short-term traffic flow prediction in bike-sharing networks. *Journal of Intelligent Transportation Systems*, 26(4):461–475.

Wang, J., Chen, R., and He, Z. (2019). Traffic speed prediction for urban transportation network: A path based deep learning approach. *Transportation Research Part C: Emerging Technologies*, 100:372–385.

Wang, Z., Su, X., and Ding, Z. (2020). Long-term traffic prediction based on lstm encoder-decoder architecture. *IEEE Transactions on Intelligent Transportation Systems*, 22(10):6561–6571.

Woodard, D., Nogin, G., Koch, P., Racz, D., Goldszmidt, M., and Horvitz, E. (2017). Predicting travel time reliability using mobile phone GPS data. *Transportation Research Part C: Emerging Technologies*, 75:30–44.

Wu, Y., Tan, H., Qin, L., Ran, B., and Jiang, Z. (2018). A hybrid deep learning based traffic flow prediction method and its understanding. *Transportation Research Part C*, 90:166–180.

Wu, Y.-J., Chen, F., Lu, C.-T., and Yang, S. (2016). Urban traffic flow prediction using a spatio-temporal random effects model. *Journal of Intelligent Transportation Systems*, 20(3):282–293.

Yang, K., Yang, H., and Du, L. (2023). A data-driven traffic shockwave speed detection approach based on vehicle trajectories data. *Journal of Intelligent Transportation Systems*, pages 1–17.

Yang, S., Ma, W., Pi, X., and Qian, S. (2019). A deep learning approach to real-time parking occupancy prediction in transportation networks incorporating multiple spatio-temporal data sources. *Transportation Research Part C: Emerging Technologies*, 107:248–265.

Yao, W. and Qian, S. (2021). From Twitter to traffic predictor: Next-day morning traffic prediction using social media data. *Transportation Research Part C*, 124:102938.

Ying, C., Chow, A. H. F., and Chin, K. (2020). An actor-critic deep reinforcement learning approach for metro train scheduling with rolling stock circulation under stochastic passenger demand. *Transportation Research Part B*, 140:210–235.

Yuan, Y., Zhang, W., Yang, X., Liu, Y., Liu, Z., and Wang, W. (2021). Traffic state classifica-

tion and prediction based on trajectory data. *Journal of Intelligent Transportation Systems*, pages 1–15.

Zhan, X., Zhang, S., Szeto, W. Y., and Chen, X. (2020). Multi-step-ahead traffic speed forecasting using multi-output gradient boosting regression tree. *Journal of Intelligent Transportation Systems*, 24(2):125–141.

Zhang, C., Ho, H., Lam, W. H., Ma, W., Wong, S., and Chow, A. H. (2022). Lane-based estimation of travel time distributions by vehicle type via vehicle re-identification using low-resolution video images. *Journal of Intelligent Transportation Systems*, pages 1–20.

Zhang, Y. and Haghani, A. (2015). A gradient boosting method to improve travel time prediction. *Transportation Research Part C: Emerging Technologies*, 58:308–324.

Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., Deng, M., and Li, H. (2019). T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, 21(9):3848–3858.

Zhong, R., Luo, J., Cai, H., Sumalee, A., Yuan, F., and Chow, A. (2017). Forecasting journey time distribution with consideration to abnormal traffic conditions. *Transportation Research Part C: Emerging Technologies*, 85:292–311.

Zhong, R., Xie, X., Luo, J., Pan, T., Lam, W. H. K., and Sumalee, A. (2020). Modeling double time-scale travel time processes with application to assessing the resilience of transportation systems. *Transportation Research Part B*, 132:228–248.

Zhu, L. and Laptev, N. (2017). Deep and confident prediction for time series at Uber. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 103–110. IEEE.