Towards a Unified Understanding of Uncertainty Quantification in Traffic Flow Forecasting

Weizhu Qian, Yan Zhao, Dalin Zhang, *Member, IEEE*, Bowei Chen, Kai Zheng, *Senior Member, IEEE*, and Xiaofang Zhou, *Fellow, IEEE*

***Abstract*—Uncertainty is an essential consideration for time series forecasting tasks. In this work, we focus on quantifying the uncertainty of traffic forecasting from a unified perspective. We develop a novel traffic forecasting framework, namely Deep Spatio-Temporal Uncertainty Quantification (DeepSTUQ), which can estimate both aleatoric and epistemic uncertainty. Specifically, we first leverage a spatio-temporal model to model the complex spatio-temporal correlations of traffic data. Subsequently, two independent sub-neural networks maximizing the heterogeneous log-likelihood are developed to estimate aleatoric uncertainty. To estimate epistemic uncertainty, we combine the merits of variational inference and deep ensembling by integrating the Monte Carlo dropout and the Adaptive Weight Averaging re-training methods, respectively. Furthermore, to relax the Gaussianity assumption, mitigate overfitting, and improve horizon-wise uncertainty quantification performance, we define a new calibration method called Multi-horizon Conformal Calibration (MHCC). Finally, we provide a theoretical analysis of the proposed unified approach based on the PAC-Bayes theory. Extensive experiments are conducted on four public datasets, and the empirical results suggest that the proposed method outperforms state-of-the-art methods in terms of both point prediction and uncertainty quantification.**

***Index Terms*—Traffic forecasting, uncertainty quantification, variational inference, deep ensembling, model calibration, PAC-Bayes.**

I. Introduction

TRAFFIC forecasting is one of the essential elements in modern Intelligent Transportation Systems (ITS). The predicted data, including but not limited to traffic flow, speed, and volume, can help municipalities manage urban transportation more efficiently. In terms of traffic forecasting, the road segments in a road network interact with each other spatially, and the current state of a road segment depends on previous states, which results in complicated spatio-temporal correlations. Modelling the spatial-temporal correlations of traffic data is non-trivial [5], [17], [30], [39], [52], [54], [57].

Thanks to the recent advances of deep learning techniques, a number of deep learning-based spatio-temporal models have been proposed in the field of traffic forecasting [44], [47]. Since the topology of a typical road network can be described by a graph in which each node represents a sensor and each edge represents a road segment, the spatial dependency of traffic data can be naturally extracted by Graph Neural Networks (GNNs) [20]. Correspondingly, the temporal dependency of traffic data can be modelled by Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), or their variants [5], [30], [39], [52].

Despite the fact that existing methods regarding traffic forecasting have been shown successful [47], most of them only provide point traffic prediction without quantifying uncertainty — a critical component in traffic data. Uncertainty quantification can be used to estimate the possible minimum and maximum values of the predicted traffic flow, speed, and volume. Such reliability information can be imperative for municipalities to manage urban traffic systems in real-world scenarios (e.g., emergency rescue and disaster evacuation) where unreliable deterministic point prediction may lead to catastrophic consequences [48]. Traffic forecasting can help improve traffic conditions and consequently reduce traffic incidents [34], [38]. However, unexpected events, e.g., accidents, will affect the accuracy of the traffic forecasting [2], [11]. Therefore, from a safety perspective, it is necessary to provide predictions with reliability information. Moreover, traffic forecasting models with uncertainty quantification can be used to develop proactive intelligent traffic control systems to prevent possible future traffic congestion.

In this paper, we aim to attain both future traffic forecasting and its corresponding uncertainty. More specifically, the research goal includes the estimation of both epistemic and aleatoric uncertainties, which refer to model uncertainty and data uncertainty, respectively. Aleatoric uncertainty can be obtained by two independent neural networks by estimating means and variances, respectively [35]. As for epistemic uncertainty, both variational inference and ensembling are possible solutions. However, these two types of approaches both have their own limitations. Variational approaches, e.g., Bayesian Neural

Manuscript received 28 November 2022; revised 9 June 2023; accepted 26 August 2023. Date of publication 6 September 2023; date of current version 5 April 2024. This work was supported in part by NSFC under Grants 61972069, 61832017, and 62272086, in part by Shenzhen Municipal Science and Technology R&D Funding Basic Research Program under Grant JCYJ20210324133607021, in part by the Municipal Government of Quzhou under Grant 2022D037, and in part by the Key Laboratory of Data Intelligence and Cognitive Computing, Longhua District, Shenzhen. Recommended for acceptance by W. Zhao. *(Corresponding author: Yan Zhao.)*

Weizhu Qian is with the School of Computer Science and Technology, Soochow University, Suzhou, Jiangsu 215006, China (e-mail: wzqian@suda.edu.cn).

Yan Zhao and Dalin Zhang are with the Department of Computer Science, Aalborg University, 9220 Aalborg, Denmark (e-mail: yanz@cs.aau.dk; dalinz@cs.aau.dk).

Bowei Chen is with the University of Glasgow, G12 8QQ Glasgow, U.K. (e-mail: bowei.chen@glasgow.ac.uk).

Kai Zheng is with the Yangtze Delta Region Institute (Quzhou), School of Computer Science and Engineering, Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Chengdu, Sichuan 610056, China (e-mail: zhengkai@uestc.edu.cn).

Xiaofang Zhou is with the Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong (e-mail: zxf@cse.ust.hk).

Digital Object Identifier 10.1109/TKDE.2023.3312261

Networks (BNNs) [21], [33], are prone to modal collapse [14]. Deep ensembling is capable of finding multiple local minimums by training a set of deterministic models, but the prediction of each trained deterministic model lacks diversity [14]. To circumvent this problem, it needs to find a set of local minimums/solutions with certain amount of diversities.

To this end, we carefully design an epistemic uncertainty quantification method integrating the merits of both variational inference and deep ensembling. Due to the high flexibility and efficiency of Monte Carlo dropout (MCDO) [12], we adopt it as the variational inference method. To implement MCDO in the proposed approach, we place the dropout operations in the base model with careful design to ensure the model performance. Despite the success of Stochastic Weight Averaging (SWA) [19] on approximating deep ensembling, we find that the original SWA method cannot guarantee the convergence of the training process of traffic forecasting tasks. In this light, we propose a new re-training method, called Adaptive Weight Averaging (AWA), to better approximate deep ensembling. Compared with SWA that uses SGD for training, AWA utilizes a new learning rate scheduler with Adam, which can approximate deep ensembling more efficiently on traffic forecasting. In addition, a post-processing calibration method is proposed to mitigate the overfitting issue in uncertainty quantification. We design the above building blocks to tackle different aspects of the uncertainty quantification problem, achieving an effective and efficient traffic forecasting framework. Finally, a unified uncertainty quantification approach called Deep Spatio-Temporal Uncertainty Quantification (DeepSTUQ) is formulated for both epistemic and aleatoric uncertainty estimation. Compared to existing approaches, DeepSTUQ has the following advantages: 1) DeepSTUQ can predict future traffic while providing both epistemic and aleatoric prediction uncertainty; and 2) DeepSTUQ requires training only one single model, which as a result, is fast-training, low-memory-footprint, and fast-inferring.

In the conference version of this work [36], Gaussian assumption was used in the calibration method. However, it may not be valid empirically with the real traffic data in many cases. The other issue of the original calibration method [36] is that it cannot guarantee horizon-wise uncertainty quantification performance. To mitigate overfitting, relax the Gaussianity assumption, and improve horizon-wise uncertainty quantification performance, we propose a novel conformal inference based calibration method, called Multi-horizon Conformal Calibration (MHCC), where the target significance can be corrected according to the proposed empirical equation. Moreover, we provide an in-depth theoretical analysis for uncertainty quantification to show that the proposed method can strengthen the point prediction performance and the horizon-wise prediction coverage performance.

The major value-added extensions over our preliminary work [36] are three-fold.

- We identify and study in depth a limitation in our previous approach, thus enabling improved horizon-wise uncertainty quantification performance.
- We propose a horizon-wise post-processing calibration method that relaxes the Gaussianity assumption and reduces overfitting, achieving better performance of uncertainty quantification in traffic flow forecasting.
- Extensive experiments are conducted on four public datasets, and the results show that the proposed DeepSTUQ advances the state of the art in terms of point prediction and uncertainty quantification.

The remainder of this paper is organized as follows. Section II surveys the related work, and Section III gives preliminary concepts. We then present the DeepSTUQ model in Section IV, followed by an empirical study in Section V. Section VI offers conclusions.

II. Related Work

Our study relates to spatio-temporal traffic forecasting regarding its application and uncertainty quantification regarding the methodology. In the following, we review the state-of-the-art methods from these two aspects in this section.

A. Spatio-Temporal Traffic Forecasting

Traffic data can be regarded as multivariate time series. Hence, for traffic forecasting tasks, both spatial and temporal correlation are critical data features to learn from. In terms of spatial correlations, Graph Neural Network, such as Graph Convolutional Networks (GCNs) [25], ChebNet [9], and Graph Attention Networks (GATs) [43], have become the *de facto* deep learning techniques. As for temporal dependency, deep architectures like Gated Recurrent Networks (GRUs) [7], Gated Convolutional Neural Networks (GCNNs) [8], and WaveNet [42], have been widely applied to traffic prediction. Base on these two types of methods, a number of deep spatio-temporal models have been proposed in the context, such as Diffusion Convolutional Recurrent Neural Network (DCRNN) [30], Temporal Graph Convolutional Network (T-GCN) [56], Spatio-Temporal Graph Convolutional Networks (ST-GCN) [54], and GraphWaveNet [52]. These methods are capable of learning spatio-temporal correlations but fail to capture multi-scale or hierarchical dependency.

More recently, Li et al. [29] proposed Spatial-Temporal Fusion Graph Neural Network (STFGNN), in which a spatial-temporal fusion graph module and a gated dilated CNN module were used to capture local and global correlations simultaneously. Zheng et al. [55] proposed Spatial-Temporal Graph Diffusion Network (ST-GDN) that adopted a hierarchical graph neural network architecture and a multi-scale attention network to learn spatial dependency from local-global perspectives and multi-level temporal dynamics, respectively. Qu et al. [37] used contrastive self-supervision to learn the spatio-temporal correlations within the coarse-grained urban traffic flows. Liang et al. [31] proposed a physics-informed approach for spatio-temporal forecasting. Additionally, the attention mechanism has been applied to address this issue as well. For instance, Attention-based Spatial-Temporal Graph Convolutional Network (ASTGCN) [17] uses spatial and temporal attention to model the spatial patterns and dynamic temporal correlations, respectively.

Nevertheless, in a practical real-world case where the knowledge of a graph is missing, the physical road connectivity may not necessarily represent the real data correlation in a graph. It is therefore beneficial to learn the graph structure from the data. To this end, methods such as Multivariate Time Series Forecasting

with Graph Neural Network (MTGNN) [51] and Adaptive Graph Convolutional Recurrent Network (AGCRN) [5] can learn the unknown adjacency matrix in a data-driven manner and consequently improve the prediction performance. However, all these aforementioned methods only focus on providing point estimation without computing prediction intervals.

B. Uncertainty Quantification

Uncertainty quantification has recently become an actively researched area and widely applied to solve various real-world problems [1], [14]. In general, uncertainty can be classified into two categories, namely, aleatoric and epistemic.

Aleatoric uncertainty refers to the data uncertainty caused by noise or intrinsic randomness of processes, which is irreducible but can be computed via predictive means and variances [22] using negative log-Gaussian likelihood as the loss function. When data distribution is unknown, distribution-free methods such as quantile regression [26] and conformal inference [4] can be used to estimate aleatoric uncertainty by computing the upper and the lower bounds of the prediction intervals with respect to the predefined significance level.

Epistemic uncertainty refers to model uncertainty caused by data sparsity or lack of knowledge, which is learnable and reducible. A widely-used method for estimating epistemic uncertainty is Bayesian Neural Networks (BNNs) [21], in which a Gaussian distribution is imposed on each weight to generate model uncertainty. However, a typical BNN doubles the number of model parameters and requires to compute the Kullback-Leibler (KL) divergence explicitly [6], which raises the model complexity and slows down the training process. Alternatively, a simple approach called Monte Carlo (MC) dropout [12] performs Bayesian approximation by turning on dropout at both training and test time as opposed to standard dropout [40].

Apart from Bayesian methods, ensembling-based approaches can be applied to uncertainty quantification as well [27], [49]. However, vanilla ensembling methods is time and memory consuming because it is required to train and store multiple models. To address this issue, Fast Geometric Ensembling (FGE) [13] and Stochastic Weight Averaging (SWA) [19] are proposed, which use varying learning rates during training to find different local minimums. In addition, model calibration methods, e.g., Temperature Scaling [16], are also used to estimate prediction uncertainty.

Although uncertainty quantification has been quite popular in many deep learning domains, such as Computer Vision [22], Medical Imaging [1] and Reinforcement Learning [14], it is less explored in traffic prediction. Wu et al. [50] analyzed different Bayesian and frequentist uncertainty quantification approaches for spatio-temporal forecasting. They figured that Bayesian methods were more robust in point prediction, whilst frequentist methods provided better coverage over ground truth variations. Other works also focus on deep learning-based spatio-temporal uncertainty quantification, e.g., [45], [46], [58], [60], [60].

In this paper, we specifically study the uncertainty quantification problem for spatio-temporal traffic prediction. The proposed approach is based on the spatio-temporal architecture [5] and combines Monte Carlo dropout, Adaptive Weight Averaging re-training, and model calibration to provide both point prediction and uncertainty estimation advancing the state of the art.

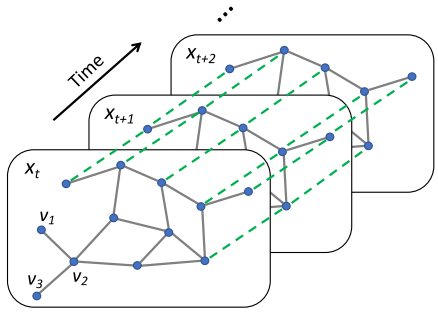


Fig. 1. Spatio-temporal dependency modelling for traffic data, where grey lines and green dash lines represent spatial and temporal dependency, respectively.

III. Problem Statement

In this section, we will describe the task of traffic forecasting and its corresponding uncertainty quantification problem.

A. Traffic Forecasting

Traffic flow data can be regarded as multivariate time series. Let $x_t \in \mathbb{R}^N$ be the values of all the sensors in a road network at time t and $X_{<t} = \{x_{t-T_h+1}, x_{t-T_h+2}, \ldots, x_t\} \in \mathbb{R}^{N \times T_h}$ be the corresponding historic input sequence with T_h steps. Similarly, $\hat{X}_{>t} = \{x_{t+1}, x_{t+2}, \ldots, x_{t+\tau}\} \in \mathbb{R}^{N \times \tau}$ represents the prediction sequence, where τ denotes the prediction horizon.

Fig. 1 describes a spatio-temporal correlation modelling problem in traffic forecasting.

Instead of treating the forecasting as deterministic, we aim to compute a conditional distribution to predict the traffic flow as well as the prediction uncertainty $\hat{X}_{>t} \sim P(\hat{X}_{>t}|X_{<t})$, which can improve the accuracy of the prediction, enhance the generalization ability of the model, and provide uncertainty estimation as well.

We consider the uncertainty assumptions on the traffic data from two aspects. The first one is the Gaussianity assumption, and the other is the distribution-free assumption.

B. Uncertainty Quantification for Traffic Forecasting

The assumptions for the predictive likelihoods in traffic forecasting can be either Gaussian or distribution-free, and we will study both in this work.

1) Gaussian Uncertainty Assumption: Although multi-modality and seasonality do exist in traffic data [17], for simplicity, we do not consider multi-modality or seasonality, and consequently treat the predictive distribution of each node at each time point as a conditional univariate Gaussian distribution. A similar assumption can also be found in a previous study [59]. To this end, $P(\hat{X}_{>t}|X_{<t})$ can be represented by a set of predictive

mean-variance pairs, and the problem can be given as follows:

$$\theta = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{N} \log \mathcal{N}(\hat{X}^i_{>t}; \hat{\mu}_\theta(X^i_{<t}), \hat{\sigma}_\theta(X^i_{<t})^2), \quad (1)$$

where θ is the model parameters, N is the number of total training data points, $\mathcal{N}$ denotes the Gaussian likelihood, and $\hat{\mu}(X_{<t})$ and $\hat{\sigma}(X_{<t})^2$ represents the estimated mean and variance, respectively.

2) Distribution-Free Uncertainty Assumption: From a distribution-free perspective, the uncertainty quantification task aims to obtain prediction interval $C_\theta(X^i_{<t}) = [\hat{y}_{L_i}, \hat{y}_{U_i}]$, such that a future ground truth datapoint $\hat{X}^i_{>t}$ falls into $C_\theta(X^i_{<t})$ with a sufficiently high probability, where $\hat{y}_{L_i}$ and $\hat{y}_{U_i}$ denote the upper and the lower bounds of the predicted interval, respectively. Let α be the significance level, and then the first optimization goal is to ensure

$$P\left(\hat{X}^i_{>t} \in C_\theta(X^i_{<t})\right) \geq 1 - \alpha, \quad (2)$$

where $P(\hat{X}^i_{>t} \in C_\theta(X^i_{<t}))$ can be any continuous probability distribution. Hence, the Gaussianity assumption aforementioned is relaxed.

Apart from satisfying (2), the width of the prediction interval should be as small as possible as well. Let q be the uncertainty scalar correspondent to α. The new prediction intervals can be rendered via using the means and variances of the learned Gaussian likelihoods. Subsequently, the upper and the lower bounds of the new constructed prediction interval are $\hat{y}_{U_i} = \hat{\mu}_i(x_i) + q\hat{\sigma}_i(x_i)$, and $\hat{y}_{L_i} = \hat{\mu}_i(x_i) - q\hat{\sigma}_(x_i)$, respectively. Accordingly, the second optimization goal is shown in the following equation.

$$\theta = \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{N} 2q\hat{\sigma}_\theta(X^i_{<t}), \quad (3)$$

where $\hat{\sigma}_\theta$ is the estimated stand deviation under the model parameters θ. Note that the distribution-free uncertainty quantification paradigm can be compatible with the Gaussianity based uncertainty quantification paradigm as it can provide adaptive un-calibrated predictive intervals, i.e., stand deviations.

IV. Deep Spatio-Temporal Uncertainty Quantification

We first briefly give an overview of the proposed Deep Spatio-Temporal Uncertainty Quantification (DeepSTUQ). The DeepSTUQ model architecture is illustrated in Fig. 2, which follows the principle of the previous study [5]. Specifically, the architecture includes an encoder and a decoder sub-neural network. The encoder is composed of a GCN and a GRU module to capture both the spatial and temporal dependencies, respectively. To estimate the aleatoric uncertainty, the decoder employs two independent convolutional layers computing means and variances, respectively. Moreover, dropout operations are deployed in both sub-networks to estimate the epistemic uncertainty.

In terms of model training, conventional training procedures are only capable of providing uni-modal solutions, which lacks diversity for quantifying uncertainty [14]. Moreover, the conventional calibration methods cannot guarantee horizon-wise prediction coverage.

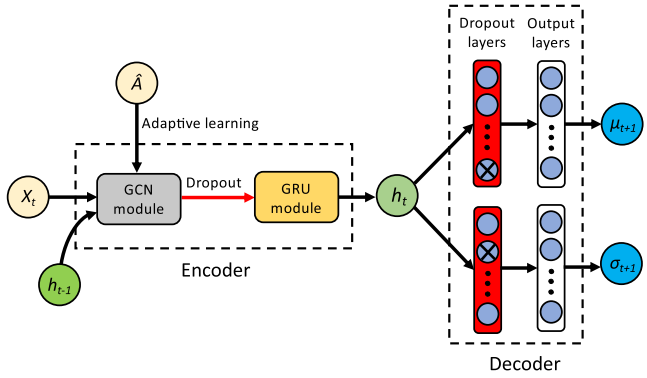


Fig. 2. Architecture of DeepSTUQ.

TABLE I
Main Variables and Their Definitions in This Paper

Variable	Definition
x_t	value of all sensors in a road network at time t
θ	model parameters
α	target significance level
lr	learning rate
lr_1, lr_2	largest & smallest learning rates of AWA
T	calibration temperature
s_c^h	nonconformality score at prediction horizon h
q_c^h	uncertainty scalar at prediction horizon h
X_{Train}	training dataset
X_{Test}	testing dataset
X_{Cali}	calibration dataset
X_{Temp}	temporary calibration dataset
N_{Cali}	calibration dataset size
N_{Update}	calibration dataset update step

To address the above issues, a four-stage uncertainty quantification approach is proposed, which can be briefed as follows.

- *Stage 1:* Pre-train the base spatial-temporal model with dropout on the training dataset to perform variational learning;
- *Stage 2:* Re-train the pre-trained model on the training dataset to proceed ensemble learning;
- *Stage 3:* Calibrate the re-trained model on the calibration dataset to further improve the aleatoric variance estimation;
- *Stage 4:* Update the calibration dataset and inference.

The major nations in this paper are listed in Table I. In the following sections, we will introduce DeepSTUQ in detail.

A. Spatio-Temporal Dependency

1) Graph Convolution: A typical road network consists of a number of road segments. The spatial relationships within a road network with N_R road segments can be described through a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where the nodes $\mathcal{V} = \{v_1, v_2, v_3, \ldots, v_{|\mathcal{V}|}\}$ denote the sensors and the edges $\mathcal{E}$ denotes the road segment.

$A \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ is the corresponding adjacency matrix. Subsequently, GCN [25] is utilized to model the spatial relationships of the traffic data. The output of the l-th GCN layer, $Z^{(l+1)}$, can be computed by

$$Z^{(l+1)} = f(Z^{(l)}, A), \tag{4}$$

where $Z^{(l)}$ is the input. More specifically, the GCN first uses a degree matrix D to avoid changing the scale of feature vectors by multiplying it with A. Afterwards, an identity matrix I is used to sum up the neighboring nodes of a node as well as the node itself. As a result, the propagation rule of the GCN is described as follows:

$$Z^{(l+1)} = S\left((I + D^{-\frac{1}{2}}AD^{-\frac{1}{2}})Z^{(l)}W^{(l)} + b^{(l)}\right), \tag{5}$$

where $W^{(l)}$ is the weight matrix, $b^{(l)}$ is the bias, and S is an activation function, e.g., a sigmoid function.

2) Graph Structure Learning: In many real-world cases, we do not have the real spatial correlation knowledge of the multivariate traffic data. In such cases, the graph structure needs to be learned from data. To this end, the adaptive learning approach proposed in the study [5] is adopted to directly generate $\hat{A} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$, which is easier than generating the adjacency matrix during the training process. Particularly, $\hat{A}$ is developed by

$$\hat{A} = \mathrm{softmax}\left(\mathrm{ReLU}(EE^T)\right), \tag{6}$$

where $E \in \mathbb{R}^{|\mathcal{V}|\times d}$ (the embedding dimension $d \ll |\mathcal{V}|$) is a learnable matrix representing the embedding of the nodes, and the $\mathrm{softmax}$ function is to normalize the learned matrix. To facilitate the graph learning process, a Node Adaptive Parameter Learning (NAPL) module [5] is also utilized to reduce the computational cost. As a result, (5) becomes

$$Z^{(l+1)} = S\left((I + \hat{A})Z^{(l)}EW_g^{(l)} + Eb_g^{(l)}\right). \tag{7}$$

Moreover, by using NAPL, the model is capable of learning the time-dependent graph structure of the traffic signals.

3) Temporal Dependency: Apart from the spatial dependency, the temporal dependency of traffic data also needs to be captured. To this end, the aforementioned graph convolutional operations and the adaptive graph learning module are integrated into a Gated Recurrent Unit (GRU) [7]. Subsequently, the obtained spatio-temporal model can be formulated as follows:

$$z_t = S\left((I + \hat{A})[x_t, h_{t-1}]EW_z + Eb_z\right), \tag{8a}$$

$$r_t = S\left((I + \hat{A})[x_t, h_{t-1}]EW_r + Eb_r\right), \tag{8b}$$

$$c_t = \tanh\left((I + \hat{A})[x_t, r_t \odot h_{t-1}]EW_c + Eb_c\right), \tag{8c}$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot c_t, \tag{8d}$$

where z stands for the update gate, r stands for the reset gate, h denotes the hidden state, $[\cdot]$ denotes the concatenation operation, c denotes the memory cell, and W and b represent the weights and bias, respectively.

Finally, the model introduced in (8) serves as the spatio-temporal architecture in DeepSTUQ. Note that though the above base model is employed in this work, DeepSTUQ has the potential to be applied to other spatial-temporal structures as well. In the following sections, we explain how to leverage this base model to forecast traffic and quantify the corresponding forecasting uncertainty.

B. Uncertainty Quantification

Generally, uncertainty can be classified into two types, i.e., epistemic and aleatoric. The former represents model uncertainty, while the latter represents data uncertainty. If variance is used to render uncertainty, the total uncertainty can be decomposed and approximated as follows:

$$\sigma^2_{\text{Total}} \approx \underbrace{\mathbb{E}_{\theta\sim p(\theta)}[\sigma^2_\theta]}_{\text{Aleatoric uncertainty}} + \underbrace{\mathbb{V}_{\theta\sim p(\theta)}[\mu_\theta]}_{\text{Epistemic uncertainty}}, \tag{9}$$

where $p(\theta)$ stands for a probability distribution over the model parameters θ, and σ^2_θ and μ_θ refer to the predicted variance and mean, respectively.

1) Aleatoric Uncertainty: Aleatoric uncertainty is caused by the intrinsic randomness of data, which is irreducible but learnable [1]. Based on (9), we assume that the lower and upper bounds of the forecasting are symmetric due to the regressive nature of the prediction. Subsequently, the distribution of a sensor's value, e.g., traffic flow, at each time point can be modeled by a Gaussian distribution with predicted mean $\mu(x)$ and variance $\sigma(x)^2$. However, directly maximizing the predictive Gaussian likelihood is numerically unstable. Instead, we choose to maximize the following log-likelihood:

$$\begin{aligned}&\log\ p(y|\mu(x), \sigma(x))\\ &= -\frac{1}{2}\log(\sigma(x)^2) - \frac{1}{2}\log(2\pi) - \frac{(y-\mu(x))^2}{2\sigma(x)^2},\end{aligned} \tag{10}$$

where $\log(\sigma(x)^2)$ and $\mu(x)$ are obtained directly via two independent neural networks.

In practice, to accelerate the training process and ensure convergence, we devise the following weighted loss by adding an L1 loss as the regularization term based on (10):

$$\begin{aligned}\mathcal{L}_{\text{Aleatoric}} = \frac{1}{N}\sum_{i=1}^{N} &\lambda\left\{\log(\sigma(x_i)^2) + \frac{(y_i - \mu(x_i))^2}{\sigma(x_i)^2}\right\}\\ &+ (1-\lambda)|y_i - \mu(x_i)|,\end{aligned} \tag{11}$$

where λ is the relative weight with $0 < \lambda \leq 1$.

2) Epistemic Uncertainty: Epistemic uncertainty represents model uncertainty, which arises from that lack of data or model mis-specification. Fortunately, as opposed to aleatoric uncertainty, epistemic uncertainty can be reduced by estimation. There are two general classes of approaches to do so: Bayesian variational inference and deep ensembling. However, they both have their pros and cons. Fig. 3 illustrates the relationships between different solutions and corresponding model performance. The solid and dashed lines represent the model performance during training and testing processes, respectively. The green line and blue dots represent the performance that can be obtained by variational inference and deterministic model, respectively. As

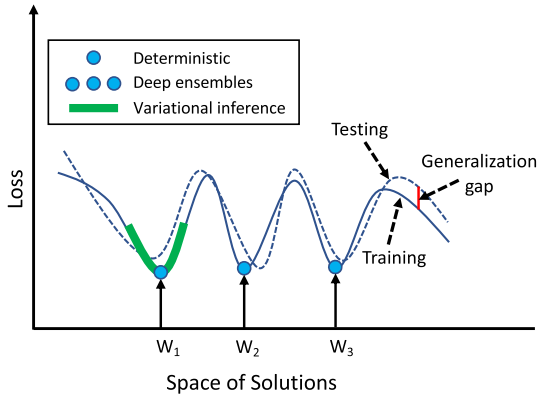


Fig. 3. Performance demonstration of deterministic model, deep ensembles, and variational inference in solution space.

it can be seen from the figure, deep ensembling can find a set of different deterministic model parameters (local minimums), e.g., W_1, W_2, and W_3, which may have equally good performance in the solution space [10]. On the other hand, variational inference can find a set of sub-optimal solutions near one local minimum in the loss space. However, it may fail to find other local minimums, which potentially leads to modal collapse. Therefore, a better way is to explore as many as local minimums as well as their corresponding nearby solutions. To this end, we propose to combine deep ensembling and variational inference to estimate epistemic uncertainty.

Variational Inference: Let $D = \{X, Y\}$ be the training dataset. From a Bayesian perspective, we assume that each weight parameter of the neural network w obeys a probabilistic distribution to represent model uncertainty, e.g., Gaussian distribution. However, in practice, the true posterior of the the neural network weights $p(w|D)$ is intractable. Therefore, a variational distribution $q(w)$ is used to approximate $p(w|D)$. Accordingly, the optimization goal is to minimize the following Kullback-Leibler (KL) divergence:

$$\begin{aligned} &D_{\mathrm{KL}}(q(w)||p(w|D)) \\ &= \int q(w) \log \frac{q(w)}{p(w)p(D|w)} dw, \\ &= D_{\mathrm{KL}}(q(w)||p(w)) - \mathbb{E}_{w\sim q(w)}[\log p(D|w)], \end{aligned} \tag{12}$$

where $p(w)$ is the prior and $\log p(D|w)$ is the predictive log-likelihood.

To solve (12), MC dropout [12] is adopted as it performs Bayesian approximation in a simple and flexible manner. The variational distribution $q(w)$ formulated in MC dropout can be described as follows. Let W_i be a matrix of shape $K_j \times K_{j-1}$ for layer i, we have

$$q(W_i) = M_i \cdot (\mathrm{diag}[z_{i,j}]_{j=1}^{K_i}), \tag{13a}$$

$$z_{i,j} \sim \mathrm{Bernoulli}(p_i) \tag{13b}$$

where W_i denotes the masked weight matrices, p_i is the dropout rate used in both training and testing processes (as opposed to standard dropout), M_i is the parameters of the neural network in the i-th layer, and $z_{i,j}$ is a binary variable indicating whether unit j at layer $i-1$ (as the input of layer i) is dropped. As a result, minimizing (12) is equivalent to minimizing the following loss function:

$$\begin{aligned} \mathcal{L}_{\mathrm{Dropout}} &= \mathbb{E}_{w\sim q(w)} E[Y, f_W(X)] + D_{\mathrm{KL}}(q(w)||p(w)), \\ &\approx \frac{1}{N}\sum_{i=1}^{N} E(y_i, f(x_i, w_i)) + \frac{\lambda_W}{2p_i}||w_i||^2, \end{aligned} \tag{14}$$

where E is the loss function, e.g., Root Mean Squared Error (RMSE) or Mean Absolute Error (MAE), λ_W is the weight decay, and $\frac{\lambda_W}{2p}||w||^2$ can be computed through applying the L2 regularization during the training process.

In terms of implementation, dropout operations are deployed at two places within the spatial-temporal model: the graph convolutional layers in the encoder and the dropout convolutional layers in the decoder. Therefore, (7) becomes

$$Z^{(l+1)} = \mathrm{sigmoid}\left(M \odot \left((I + \hat{A})Z^{(l)}EW_g^{(l)} + Eb_g^{(l)}\right)\right). \tag{15}$$

Note that the dropout rate here should be small when the adjacency matrix dimension is small, and vice versa.

Combined Uncertainty: Finally, (11) and (14) are combined to estimate both aleatoric and epistemic uncertainty jointly. The combined loss function is formulated by

$$\begin{aligned} \mathcal{L}_{\mathrm{Combined}} = \frac{1}{N}\sum_{i=1}^{N} &\lambda \left\{\log(\sigma(x_i)^2) + \frac{(y_i - \mu(x_i))^2}{\sigma(x_i)^2}\right\} \\ &+ (1-\lambda)|y_i - \mu(x_i)| + \frac{\lambda_W}{2p}||w||^2. \end{aligned} \tag{16}$$

Equation (16) is utilized to pre-train the spatio-temporal model in DeepSTUQ.

Deep Ensembling: In contrast to variational inference, deep ensembling aims to find a set of different local minimums and averages the output of each trained model as the final prediction. Deep ensembling is shown to be quite effective in practice, yet it is computationally expensive as multiple models are trained [27]. FGE [13] tackles this issue by using cycling learning rate to produce a set of different trained models in one learning process. However, FGE still needs to store multiple models for inference, which may result in high memory cost. To address this issue, SWA [19] adjusts the learning rate and averages the weights during the learning process to generate only one trained model to approximate FGE. In SWA, the model parameters are updated by

$$w_{\mathrm{SWA}} = \frac{w_{\mathrm{SWA}} \cdot n_{\mathrm{models}} + w}{n_{\mathrm{models}} + 1}, \tag{17}$$

where w_{SWA} is the parameters of the SWA model and n_{models} is the number of averaged models during training.

Inspired by SWA, we devise a re-training method called Adaptive Weight Averaging (AWA) to approximate deep ensembling. As depicted in Fig. 4, we vary the learning rate during the re-training process to find different local minimums and average those local minimums in the final stage to attain better solutions. The proposed AWA re-training approach includes two steps. Let

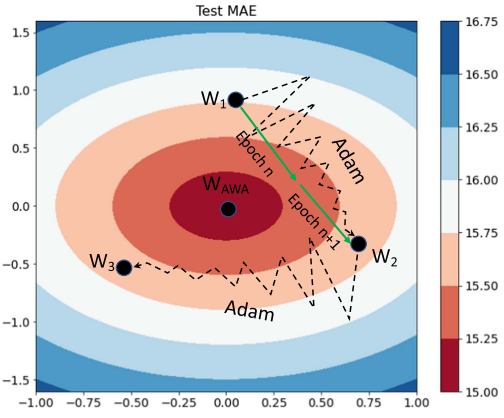


Fig. 4. Demonstration of relationship between test MAEs and model weights during AWA re-training.

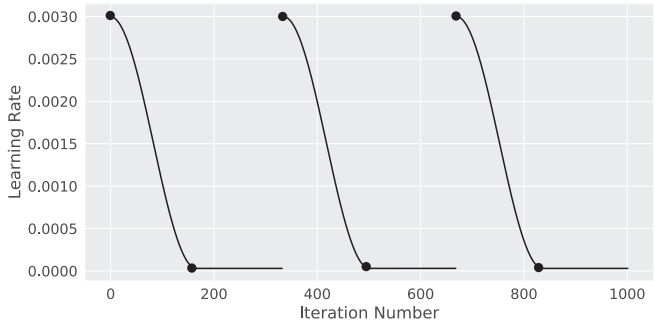


Fig. 5. Learning rate change during the AWA re-training, each black dot indicates the start of a new epoch.

the re-training learning rate be lr, the maximum learning rate be lr_1, the minimum learning rate be lr_2, and $n_{\text{iteration}}$ be the total iteration number within each epoch/total batch number, then the learning rate at the n_i-th iteration changes according to the following rules. The first step is to enable the trained model to escape from the current local minimum. To this end, the learning rate of the optimizer decreases from lr_1 to lr_2 via a cosine learning rate scheduler at epoch n. The scheduler is described by

$$lr = lr_2 + \frac{1}{2}(lr_1 - lr_2)\left(1 + \cos\left(\frac{n_{\text{iteration}}}{n_i}\pi\right)\right). \quad (18)$$

Following that, the model is fine tuned by using the constant learning rate lr_2 at epoch $n+1$, then at the end of the epoch the model parameters are averaged according to (17) and perform batch normalization. Specifically, we find that in practice using Adam [24] as the optimizer works more effectively than using Stochastic Gradient Decent (SGD) which is adopted in the original SWA method. In terms of finding different local minimums, Adam escapes saddles points where the gradients are close to 0 more efficiently than SGD, but these minimums found are sharp ones [53]. Therefore we average those sharp minimums to obtain the flat minimum to finally achieve better generalization [23]. The learning rate change during the AWA re-training is illustrated in Fig. 5. The whole re-training process is summarized in Algorithm 1.

Algorithm 1: AWA Re-Training Method.

Require: training dataset $\{X\}$; pre-trained model parameters w; AWA model parameters w_{AWA}; learning rates lr_1 and lr_2; total epoch $\text{epoch}_{\text{AWA}}$; total iteration/batch number $n_{\text{iteration}}$.

while epoch $<$ $\text{epoch}_{\text{AWA}}$:
  while $n < n_{\text{iteration}}$:
    compute the loss function according to (16) and update w;
    if epoch $\bmod 2 = 0$:
      lr decreases from lr_1 to lr_2 according to (18);
    else: $lr = lr_2$;
  end while
  if epoch $\bmod 2 = 0$ and epoch $\neq 0$:
    update w_{AWA} according to (17);
    perform batch normalization.
end while
Return w_{AWA}

For testing, we quantify the epistemic uncertainty by drawing multiple Monte Carlo samples from the learnt posterior distribution, then use the means and variances of the samples as the predictive mean and variances, respectively.

3) Model Calibration: To prevent the uncertainty estimation of the trained models being overconfident with respect to the training dataset, it is necessary to calibrate the trained model on the held-out validation/calibration dataset via post-processing.

To do so, we propose two calibration methods. One is based on the Gaussian likelihood assumption, and the other is based on the distribution-free assumption. See Section III.

Temperature Scaling Calibration: If the Gaussian likelihood assumption is still assumed to be held, a positive learnable variable T is imposed on the learned variance. Subsequently, the following log-likelihood similar to (10) is maximized:

$$\begin{aligned}
&\log p(y|\mu(x), \sigma(x)/T) \\
&= \log\left(\frac{T}{\sigma(x)\sqrt{2\pi}} e^{-\frac{1}{2}\left(\left(\frac{T(y-\mu(x))}{\sigma(x)}\right)^2\right)}\right) \\
&= \log\left(\frac{T}{\sigma(x)\sqrt{2\pi}}\right) + \log\left(e^{-\frac{1}{2}(\frac{T(y-\mu(x))}{\sigma(x)})^2}\right) \\
&= -\frac{1}{2}\log\left(\frac{\sigma(x)^2}{T^2}\right) - \frac{1}{2}\log(2\pi) - \frac{T^2\left(y-\mu(x)\right)^2}{\sigma(x)^2} \\
&= \frac{1}{2}\log(T^2) - \frac{1}{2}\log\left(\sigma(x)^2\right) - \frac{T^2(y-\mu(x))^2}{2\sigma(x)^2} - \frac{1}{2}\log(2\pi),
\end{aligned} \quad (19)$$

where T is the only learnable parameter. Accordingly, the calibration objective is

$$T = \underset{T}{\text{argmin}} \frac{1}{N_{\text{Cali}}} \sum_{i=1}^{N} -\log(T^2) + \frac{T^2(y_i - \mu(x_i))^2}{\sigma(x_i)^2}, \quad (20)$$

where $\mu(x_i)$ and $\sigma(x_i)^2$ can be obtained via one deterministic forward pass or Monte Carlo estimation. Limited-memory Broyden Fletcher Goldfarb Shanno algorithm (L-BFGS) is used as the optimizer to find the optimal value of T.

Multi-Horizon Conformal Calibration: Time series forecasting, e.g., traffic forecasting, may not rigorously comply with conditional Gaussian assumption on the predictive likelihood. Hence, it is necessary to relax the Gaussianity assumption to quantify the uncertainty in a distribution-free fashion. Furthermore, it can be found that for multi-horizon forecasting, the predicted intervals obtained at farther horizons may undercover the future ground truth datapoints. This makes the uncertainty quantification performance less reliable. Therefore, to address the above two issues, we propose the Multi-horizon Conformal Calibration (MHCC) approach via a split conformal inference fashion [28] to calibrate the trained model.

As a conformal inference method, MHCC calibrates the trained model by computing the quantiles without proceeding any optimization procedure, which makes it fast-computing. To obtain the quantile, we first need to compute the nonconformity scores to rank the prediction residuals. We do not use the absolute prediction error, i.e., $|\hat{x}_i - x_i|$, to compute the nonconformity scores as it is not adaptive. Instead, we leverage the predicted means and variances obtained by the Gaussian predictive likelihoods to compute nonconformity scores. The advantage of doing so is that we do not need to train an auxiliary model to calculate the prediction residuals. MHCC does not adopt the Bonferroni correction [41] to obtain corrected significance level α_c as its assumption is too conservative ($\alpha_c \leq \alpha/\tau$), which results in overlarge predictive intervals.

Instead, a novel significance level correction method is proposed in MHCC, which can be described as follows. We first assume that the Gaussianity assumption holds for each horizon h ($h \in \{1, 2, \ldots \tau\}$), i.e., if α is set to 0.05, then $q = 1.96$. As a result, the empirical horizon-wise prediction interval coverage rate on the calibration dataset p_c on horizon h can be computed by

$$p_c^h = \frac{1}{N_{\text{Cali}}} \sum_{i=1}^{N} k_i^h, \tag{21}$$

where N_{Cali} is the number of datapoints of the calibration dataset. Next, $k_i^h = 1$ if $\hat{X}_i^h \in C(X_i^h)$; otherwise, $k_i^h = 0$.

Then, the empirical significance level for each horizon is $1 - p_c^h$. Afterwards, to reach the ideal significance level, $1 - p_c^h$ needs to be corrected to α. To this end, the corrected horizon-wise significance level α_c on horizon h can be attained by the following empirical equation.

$$\alpha_c^h = (p_c^h + 2\alpha - 1) + \gamma(p_c^1 - p_c^H)(h-1)^2, \tag{22}$$

where γ is a positive scalar. The first term, $p_c^h + 2\alpha - 1$, is to compute the corrected significance level according to the empirical PICP. The second term, $\gamma(p_c^1 - p_c^H)(h-1)^2$, represents the time-decay, where $p_c^1 - p_c^H$ and $(h-1)^2$ are the data-dependent and horizon-dependent decay scalars, respectively.

Afterwards, the new horizon-wise uncertainty scalar q_h can be obtained using conformal inference. Therefore, for the h-th horizon, the new upper and lower bounds become $\hat{y}_{U_i^h} = \hat{\mu}_i^h + q_h \hat{\sigma}_i^h$ and $\hat{y}_{L_i^h} = \hat{\mu}_i^h - q_h \hat{\sigma}_i^h$, respectively. The MHCC approach is summarized in Algorithm 2.

Algorithm 2: MHCC Method.

Require: calibration dataset X_c; trained model f_w; significance level: $\alpha = 0.05$; uncertainty scalar $q = 1.96$; number of calibration datapoints: N_{Cali}; horizon: $h = 0$; number of prediction horizons: τ.

```
initialize μ̂_c = {}, σ̂_c = {}, and nonconformity scores s_c = {}, and calibrated horizon-wise significance level q_h = {};
while inference:
    μ̂_i, σ̂_i ← f_w(x_c^i);
    μ̂_c ← μ̂_c ∪ μ̂_i, σ̂_c ← σ̂_c ∪ σ̂_i;
end while
while h < τ:
    compute p_c^h based on μ̂^h, σ̂^h, and ground truth x^i_{>t};
    compute α_c^h using (22);
    while i < N_Cali:
        s_c^h ← s_c^h ∪ |μ̂_i^h − y_i^h| / σ̂_i^h;
        q_c^h ← [(N_Cali + 1)(1 + α_c^h)]-th smallest residual in s_c^h compute new horizon-wise uncertainty scalar.
    h = h + 1.
    end while
    q_c = q_c ∪ q_c^h
end while
Return q_c
```

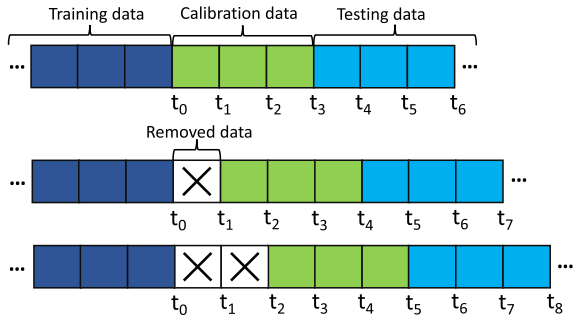


Fig. 6. Visualization of data splitting in Online MHCC.

Furthermore, conformal inference relies on i.i.d. assumption which may become weaker as time pass for time series data. To address this issue, we propose the Online MHCC by updating the calibration in an online fashion, which is illustrated in Fig. 6. Finally, the Online MHCC approach is summarized in Algorithm 3.

C. Unified Approach

Finally, combining the spatio-temporal correlation modelling method, Monte Carlo dropout, AWA re-training, and model calibration, the pipeline of the proposed unified uncertainty

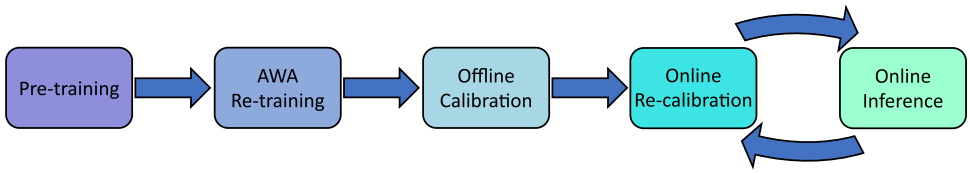


Fig. 7. Pipeline of the proposed method.

Algorithm 3: Online MHCC Method.

Require: testing dataset X_{Test}; calibration dataset X_{Cali}; trained model f_w; significance level: $\alpha = 0.05$; initial uncertainty scalar q_c; step: $i = 0$; calibration dataset update size: N_{Update}.

initialize temporary calibration dataset $X_{\text{Temp}} = \{\}$;
while testing:
  inference using f_w, X_i, and q_c;
  $i = i + 1$;
  $X_{\text{Temp}} \leftarrow X_{\text{Temp}} \cup X_i$;
  if $i \bmod N_{\text{Update}} = 0$;
    update X_{Cali} with X_{Temp};
    calculate empirical PICP on X_{Cali};
    re-correct q_c with X_{Cali} using (22);
    reset $X_{\text{Temp}} = \{\}$.
end while

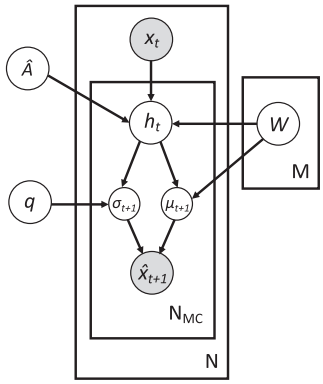


Fig. 8. Graphical model representation of DeepSTUQ, where shaded circles represent observable variables, arrows denote dependencies, variables within rectangles appear repeatedly, N_{MC} is the number of Monte Carlo samples, and M is the number of models for ensembling.

quantification method is shown in Fig. 7, which can be summarized as follows.

- First, the spatio-temporal model introduced in Sections IV-A and IV-A3 is pre-trained using (16) as the training loss function on the training dataset to estimate the aleatoric and epistemic uncertainty.
- Afterwards, the pre-trained model is re-trained via the AWA method on the training dataset to approximate deep ensembling.
- Finally, the predicted σ^2 obtained via the re-trained model on the validation dataset is calibrated according to (20).

The graphical probabilistic model representation of DeepSTUQ is visualized in Fig. 8. The figure shows that h_t is extracted from x_t via a spatio-temporal structure with a learnable variable $\hat{A}$. The model weights are drawn repeatedly to estimate the epistemic uncertainty, which is implemented in an efficient manner by using MC dropout and AWA. The variance $\sigma(x_i)^2$ and mean $\mu(x_i)$ are obtained via N_{MC} Monte Carlo samples. Finally, $\sigma(x_i)^2$ is calibrated through learning an auxiliary variable q.

For testing, according to (9), we draw N_{MC} Monte Carlo samples to estimate the predictive mean $\hat{\mu}_{t+1}$ and variance $\hat{\sigma}^2_{t+1}$ by

$$\hat{\mu}_{t+1} = \frac{1}{N_{MC}} \sum_{j=1}^{N_{MC}} \mu^j(x_t), \tag{23a}$$

$$\hat{\sigma}^2_{t+1} = q^2 \sum_{j=1}^{N_{MC}} \frac{\sigma^j(x_t)^2}{N_{MC}} + \sum_{j=1}^{N_{MC}} \frac{\left(\mu^j(x_t) - \hat{\mu}_{t+1}\right)^2}{N_{MC} - 1}, \tag{23b}$$

where $\hat{\mu}_{t+1}$ is used as the point prediction of the proposed approach, and $q^2 = \frac{1}{T}$ when using the TS calibration method.

D. Theoretical Analysis: A PAC-Bayesian Perspective

We adopt the Probabilistic Approximate Correct (PAC)-Bayes theory [3], [32] to explain the proposed method. To this end, we first assume that the datapoints in the training, calibration, and testing datasets are all i.i.d. Once the model architecture is specified, we have the hypothesis/parameter space $\mathcal{H}$. Consequently, the learning goal is to obtain a hypothesis $h \sim \mathcal{H}$. Let D be some unknown data distribution over $\mathcal{X} \times \mathcal{Y}$ and $\mathcal{L} : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \to \mathbb{R}$ be the loss function. Then true risk is defined as follow:

$$R(h) = \mathbb{E}_{(x,y)\sim D}\left[\mathcal{L}\left((x,y),h\right)\right]. \tag{24}$$

Pre-Training: In practice, $R(h)$ cannot be computed as we do not know the distribution of D. Instead, we have the training dataset $D_{\text{Train}} \sim D$. Therefore, $R(h)$ is estimated by computing the following empirical risk:

$$r(h) = \frac{1}{N_{\text{Train}}} \sum_{(x,y)\sim D_{\text{Train}}} \left[\mathcal{L}\left((x,y),h\right)\right]. \tag{25}$$

Since our problem is a regression task with a Gaussian likelihood loss function, the parameter space $\mathcal{H}$ has to be restricted to only consider variances higher than a given constant [15]. Then we have the following theorem.

Theorem 1: For all probability measure Q supported on $\mathcal{H}$, with at least probability of $1 - \delta$, the following PAC-Bayes

bound holds [15]:

$$R(h) \leq r(h) + \frac{1}{N_{\text{Train}}} \left(D_{\text{KL}}(Q||P) + \log(1/\delta)\right) + \text{const}, \tag{26}$$

where N_{Train} is the number of training datapoints. The above PAC-Bayes bound is minimized as the $D_{\text{KL}}(Q||P)$ is minimized. Hence, we do not minimize the PAC-Bayes bound such that the KL divergence does not need to be computed explicitly. Instead, we minimize the evidence lower bound (ELBO) as in (12) using Mote Carlo dropout.

Re-Training. Furthermore, note that unlike Bayesian learning, PAC-Bayes does not need to know the exact form of the prior. Thus, we can formulate a delicate posterior by combining variational inference and deep ensembling to obtain better generalization through training. To this end, a mixture of Dirac-delta distributions is constructed:

$$q(h|D_{\text{Train}}) = \frac{1}{N_M} \sum_{i=1}^{N_M} \delta(h|D_{\text{Train}}), \tag{27}$$

where $q(h|D_{\text{Train}})$ is the mixed Dirac delta posterior distribution over the model parameters. The PAC-Bayes bound (see (26)) holds for all $q \sim Q(\mathcal{H})$ in a set of Dirac masses $\{\delta \sim h \in \mathcal{H}\}$. In the proposed method, AWA is used to generate a mixture of Dirac-delta distributions sequentially in a computationally efficient manner. By approximating Bayesian model averaging via AWA, we can attain wider minima, which consequently leads to better generalization [19], [23].

Offline Calibration: The proposed calibration method is in a split conformal fashion. Once the training process is finished, a fixed hypothesis h is rendered. Let $D_{\text{Cali}} \sim D$ be a held-out calibration dataset, and h is independent of D_{Cali}. Then, we have the theorem as follows.

Theorem 2: With at least probability of $1 - \delta$, the following PAC bound holds [18]:

$$R(h) \leq r_{D_{\text{Cali}}}(h) + \sqrt{\frac{1}{2N_{\text{Cali}}} \log(2/\delta)}, \tag{28}$$

where N_{Cali} is the number of calibration samples. The above bound suggests that the true risk can be approximated by validation using sufficient number of datapoints. We can see that the proposed calibration method based on conformal inference is essentially attained via this validation PAC bound.

Online Re-Calibration: For time series data, the i.i.d. assumption may become weaker as time passes, which is detrimental to conformal methods. Therefore, the i.i.d. assumption can be re-enforced by updating the calibration dataset such that the validation PAC bound (cf. (28)) will continue to hold during inference.

V. Experiments

To compare the performance of DeepSTUQ with other state-of-the-arts, extensive experiments are conducted on real-world datasets in terms of point prediction, uncertainty quantification, and ablation study.

In terms of the hardware environment, the CPU and GPU used for the experiments are AMD EPYC 7302 and NVIDIA Tesla T4, respectively. And for the software environment, all the methods are implemented via Python, CUDA 10.6, and Pytorch 1.7.

A. Datasets

Four different public datasets collected from the Caltrans Performance Measurement System (PEMS), i.e., PEMS03, PEMS04, PEMS07, and PEMS08 [39] are used for evaluation. We believe this type of data can be used to access the performance our approach on real-world traffic flow forecasting tasks and for fair comparison with other state-of-the-art methods.

The time interval of the traffic flow data is 5 minutes. For each prediction, we use the historic data of one-hour time range (12 time steps) as the input to predict the future traffic data of one-hour time range (12 time steps). All the datasets are split into three parts with ratio $6:2:2$ for training, validation/calibration, and testing, respectively.

B. Settings

Pre-Training: The total number of training epochs is 100. The optimizer is Adam with learning rate 0.003 and weight decay 10^{-6}. The batch size is 64. The relative weight λ in (11) for computing the aleatoric uncertainty is 0.1. The dropout rates of the graph convolutional operations in the encoder are 0.1 for PEMS03, PEMS04, and PEMS07 (the adjacency matrice are relatively large), and 0.05 for PEMS08 (the adjacency matrix is relatively small). The dropout rate at the final dropout layer in the decoder for all the datasets is 0.2.

AWA Re-Training: The optimizer of the AWA re-training process is Adam, and the maximum and minimum learning rates are 0.003 and 0.00003, respectively. The total number of re-training epochs is 20, which means that ten models are averaged.

Model Calibration: The number of Monte Carlo samples for calculating σ^2 is 10. In TS, the steps and numbers of iterations of the L-BFGS optimizer are 0.02 and 500, respectively. In offline and online MHCC, γ is set to 0 for PEMS03 and 0.03 for PEMS04, PEMS07, and PEMS08. The number of datapoints for online updating is 1,000.

Inference: To balance the inference time and model performance, we generate ten Monte Carlo samples for Bayesian model averaging.

C. Baselines

To compare the proposed DeepSTUQ with the state-of-that-art methods on point prediction and uncertainty quantification, two groups of recent traffic prediction methods are adopted as the baselines, respectively.

1) Point Prediction Baselines:

- *DCRNN* [30] adopts diffusion convolution and sequence-to-sequence learning.

- GraphWaveNet (*GWN*) [52] adopts a self-adaptive adjacency matrix and dilated casual convolution.
- *ST-GCN* [54] utilizes a GNN and a GCNN to forecast traffic.
- *ASTGCN* [17] employs Attention mechanism to model spatio-temporal dependency.
- *STSGCN* [39] forecasts traffic by synchronously extracting spatial-temporal correlations.
- *STFGNN* [29] employs a spatial-temporal fusion module and a gated dilated CNN.
- *AGCRN* [5] leverages a Node Adaptive Parameter Learning module and a Data Adaptive Graph Generation module to enhance traffic prediction performance.
- *DeepSTUQ/S* refers to the proposed method with single deterministic forward pass (dropout is turned off in testing).

2) Uncertainty Quantification Baselines: Representative approaches of different uncertainty estimation paradigms (namely, frequentist, quantile prediction, Bayesian, and ensembling) are used as the baselines. Note that all the following methods employ the same base model structure for fair comparison.

- *Point* prediction refers to the AGCRN model which is used here to compare with other uncertainty quantification methods.
- *Quantile* regression [26] is a distribution-free method which directly computes the mean, lower and upper bounds using the corresponding quantile (0.025, 0.5, 0.975).
- Mean Variance Estimation (*MVE*) [35] refers to the method that estimates heterogeneous aleatoric uncertainty through computing (11).
- Monte Carlo dropout (*MCDO*) [12] performs dropout at both training and test time, where the number of Monte Carlo samples for inference is 10.
- *Combined* refers to the method that calculates both epistemic and aleatoric uncertainty using (16) [22], where the number of Monte Carlo samples for inference is 10.
- Temperature Scaling (*TS*) [16] calibrates the aleatoric uncertainty obtained by MVE.
- Fast Geometric Ensembling (*FGE*) [13] performs fast ensembling via varying the learning rate, where the number of the stored trained models is 10.
- Locally Weighted *Conformal* Inference [4], [28] calibrates the aleatoric uncertainty obtained by MVE via conformalization.
- Conformal Forecasting Recurrent Neual Network (*CFRNN*) [41] computes the multi-horizon uncertainty using conformal prediction.

D. Metrics

Two groups of metrics are employed to evaluate the point prediction and uncertainty quantification performance, respectively.

1) Point Prediction Metrics: The point traffic forecasting performance are evaluated by the following metrics.

1) Root Mean Squared Error (*RMSE*):

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}, \tag{29}$$

where y_i is the ground truth, and $\hat{y}_i$ is the prediction.

2) Mean Absolute Error (*MAE*):

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{y}_i - y_i\right|. \tag{30}$$

3) Mean Absolute Percentage Error (*MAPE*):

$$\text{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{\hat{y}_i - y_i}{y_i}\right|. \tag{31}$$

2) Uncertainty Quantification Metrics: The uncertainty quantification performance are evaluated by the following metrics.

1) Mean Negative Log-Likelihood (*MNLL*):

$$\text{MNLL} = \frac{1}{N}\sum_{i=1}^{N} -\log \mathcal{N}(y_i; \hat{\mu}_i, \hat{\sigma}_i^2), \tag{32}$$

where $\hat{\mu}_i$ and $\hat{\sigma}_i^2$ are the predicted mean and predicted variance, respectively.

2) Prediction Interval Coverage Probability (*PICP*). The predicted lower and upper bounds of the prediction interval are denoted by $\hat{y_L}$ and $\hat{y_U}$, respectively. Let the significance level α be 5%, which means that the expected probability of a ground truth data point falling into the range $[\hat{y}_L, \hat{y}_U]$ is 95% ($100\% - \alpha = 95\%$). Accordingly, under Gaussianity assumption, we have $\hat{y}_{U_i} = \hat{\mu}_i + 1.96\hat{\sigma}_i$, and $\hat{y}_{L_i} = \hat{\mu}_i - 1.96\hat{\sigma}_i$. Let k_i^j indicate whether the real speed value of a road segment j at time i is captured by the estimated prediction interval, and we have

$$k_i = \begin{cases} 1, \text{ if } \hat{y}_{L_i} \le y_i \le \hat{y}_{U_i} \\ 0, \text{ else.} \end{cases} \tag{33}$$

Then PICP can be formulated by

$$\text{PICP} = \frac{1}{N}\sum_{i=1}^{N} k_i. \tag{34}$$

Ideally, PICP should be equal or greater than 95%.

3) Mean Prediction Interval Width (*MPIW*):

$$\text{MPIW} = \frac{1}{N}\sum_{i=1}^{N} \hat{y}_{U_i} - \hat{y}_{L_i}. \tag{35}$$

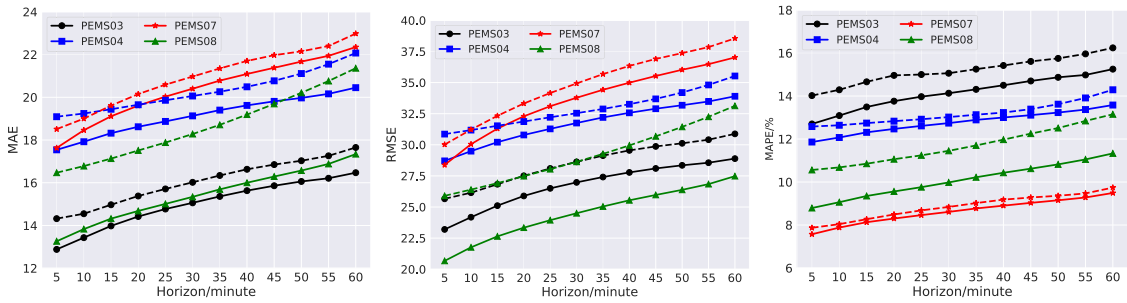
E. Point Prediction Results

The point prediction results of DeepSTUQ are compared with the aforementioned state-of-the-art methods for performance evaluation. The obtained point prediction results are demonstrated in Table II. As it can be seen from the results, with only ten Monte Carlo samples, DeepSTUQ achieves the smallest RMSEs, MAEs, and MAPEs, which suggests that DeepSTUQ has the best performance on point traffic flow prediction. In

TABLE II
POINT PREDICTION RESULTS ON PEMS03, PEMS04, PEMS07, AND PEMS08

Dataset	Metrics	DCRNN	ST-GCN	GWN	ASTGCN	STSGCN	STFGNN	AGCRN	DeepSTUQ/S	DeepSTUQ
PEMS03	MAE	18.18	17.49	19.85	17.69	17.48	16.77	16.05	15.38	**15.13**
	RMSE	30.31	30.12	32.94	29.66	29.21	28.34	28.61	27.03	**26.77**
	MAPE (%)	18.91	17.15	19.31	19.40	16.78	16.30	15.19	14.45	**14.03**
PEMS04	MAE	24.70	22.70	24.14	22.93	21.19	19.83	19.83	19.42	**19.11**
	RMSE	38.12	35.55	37.60	35.22	33.65	31.88	32.26	32.07	**31.68**
	MAPE (%)	17.12	14.59	17.93	16.56	13.90	13.02	12.97	12.98	**12.71**
PEMS07	MAE	25.30	25.38	26.85	28.05	24.26	22.07	20.94	20.76	**20.36**
	RMSE	35.58	38.78	42.78	42.57	39.03	35.80	34.98	34.12	**33.71**
	MAPE (%)	11.66	11.08	12.12	19.32	10.21	9.21	8.85	8.90	**8.63**
PEMS08	MAE	17.86	18.02	19.13	18.61	17.13	16.64	15.95	15.74	**15.44**
	RMSE	27.83	27.83	31.05	28.16	26.80	26.22	25.22	24.93	**24.60**
	MAPE (%)	11.45	11.40	12.68	13.08	10.96	10.60	10.09	10.31	**10.06**

Best and second best results are highlighted in bold and underlined, respectively.



(a) MAE (b) RMSE (c) MAPE

Fig. 9. Point prediction results with respect to various forecast horizons, where solid and dashed lines denote DeepSTUQ and AGCRN, respectively.

addition, the proposed method — even with only one single deterministic forward pass, namely DeepSTUQ/S — also outperforms other state-of-the-art methods, which indicates that the proposed method is competitive on point prediction at nearly the same inference time cost as other deterministic approaches. This is because that variational inference can obtain a set of solutions around on one local minimum, and deep ensembling can find multiple local minimums in the solution space. By combining these two approaches, DeepSTUQ is capable of finding better sub-optimal solutions and has better generalization ability compared to deterministic methods, and consequently has better performance regarding point prediction. Fig. 9 shows the point prediction performance with respect to different horizons, which suggests that DeepSTUP has better performance than AGCRN at each time step for all the datasets.

F. Uncertainty Quantification Results

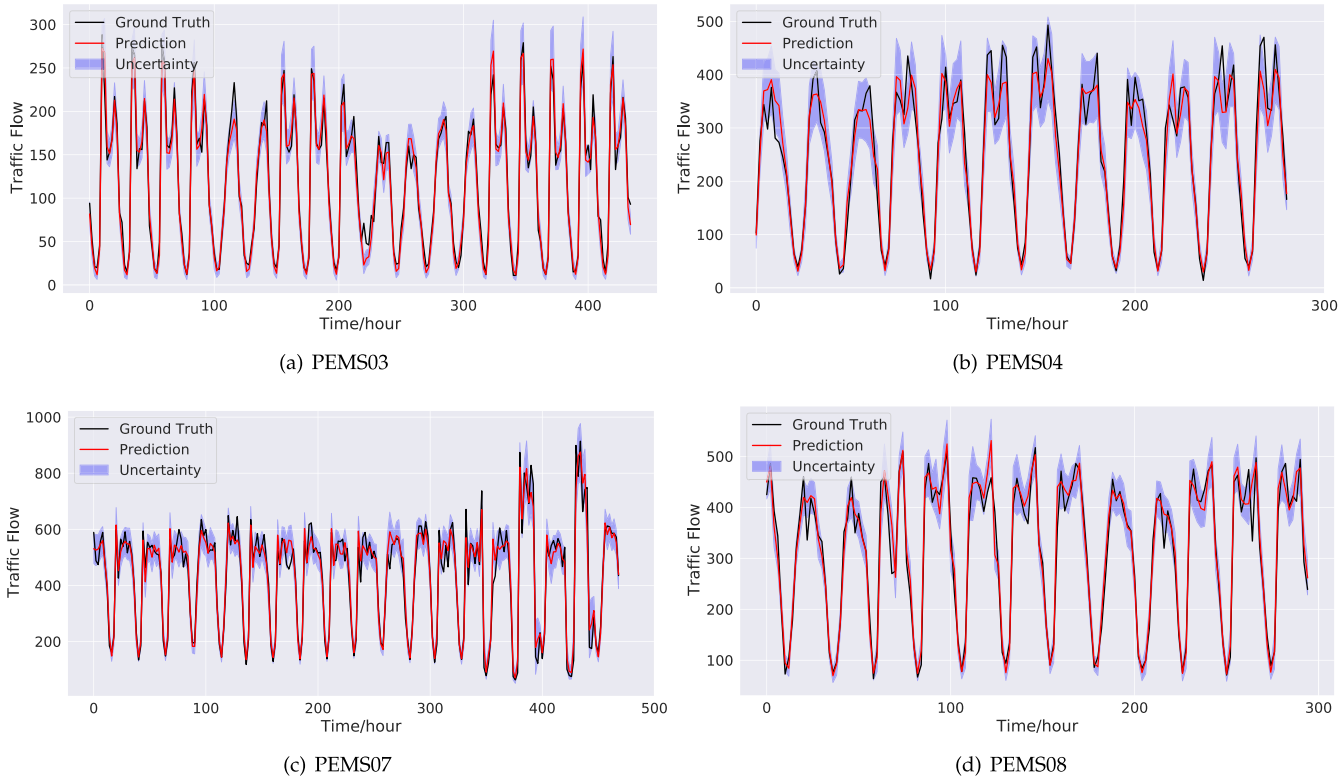
To evaluate the uncertainty quantification performance, DeepSTUQ is compared with the uncertainty quantification baselines, whose results are demonstrated in Table III and Figs. 10, 11, and 12. According to the results in the table, the proposed approach has the best overall performance regarding both the point prediction and uncertainty quantification results compared with others. As it is observed from Fig. 10, DeepSTUQ can forecast traffic flow accurately and provide valid coverage for future ground truth. Fig. 11 illustrates that in traffic flow forecasting, the aleatoric uncertainty is much larger than the epistemic uncertainty. Hence, considering total uncertainty can provide better uncertainty estimation than considering either one alone. Fig. 12 shows that, for all the datasets, generally, both aleatoric and epistemic uncertainty increase as the prediction horizons extend, which implies that short-term traffic flow forecasting is more reliable than long-term one. The conclusion accords with the intuition and results in the literature [5], [29], [30]

In terms of uncertainty quantification, the aleatoric uncertainty-aware approaches, i.e., MVE and TS, outperform the epistemic uncertainty-aware approaches, which suggests that the traffic uncertainty is mainly data-related. The results indicate that only considering epistemic uncertainty improves the estimation of the predicted mean (which results in better point estimation) but underestimates the variance significantly. This conclusion is supported by the study [50] as well. Although we have made a strong Gaussianity assumption on the likelihood of the aleatoric uncertainty, the obtained experimental results indicate that the methods using this assumption (i.e., MVE, Combined, TS, and DeepSTUQ) outperform the distribution-free method, Quantile. Additionally, the PICPs obtained by DeepSTUQ on the four datasets are very close to or larger than 95%, which implies that the Gaussian distribution assumption is credible.

TABLE III
UNCERTAINTY QUANTIFICATION RESULTS ON PEMS03, PEMS04, PEMS07, AND PEMS08, THE BEST RESULTS ARE HIGHLIGHTED IN BOLD

Dataset	Metrics	Point	Quantile	MVE	MCDO	Combined	TS	FGE	Conformal	CFRNN	DeepSTUQ
PEMS03	MAE	16.05	16.06	15.97	15.23	15.29	15.97	15.23	15.97	16.05	**15.13**
	RMSE	28.61	28.40	28.17	26.95	27.13	28.17	26.99	28.17	28.61	**26.77**
	MAPE(%)	15.19	15.50	15.08	14.39	14.60	15.08	14.36	15.08	15.19	**14.03**
	MNLL	–	–	3.53	12.32	3.39	3.49	25.94	3.53	–	**3.38**
	PICP(%)	–	89.49	92.06	43.92	93.64	93.51	31.15	93.21	93.00	**94.98**
	MPIW	–	65.60	74.04	19.73	73.26	79.79	12.81	76.72	82.79	79.86
PEMS04	MAE	19.83	20.08	19.86	19.15	19.23	19.86	**19.08**	19.86	19.83	19.11
	RMSE	32.26	32.76	32.30	**31.49**	31.73	32.30	31.59	32.30	32.26	31.68
	MAPE(%)	12.97	13.06	13.25	12.77	12.87	12.97	**12.69**	13.25	12.97	12.71
	MNLL	–	–	3.71	23.17	3.63	3.70	15.47	3.71	–	**3.57**
	PICP(%)	–	91.87	93.10	34.18	**95.16**	94.86	42.60	94.33	94.65	**95.00**
	MPIW	–	91.72	102.97	17.30	108.44	115.48	21.95	109.74	118.83	**103.76**
PEMS07	MAE	20.94	21.28	21.07	20.61	20.37	21.07	20.42	21.07	20.94	**20.36**
	RMSE	34.98	35.76	34.94	34.20	**33.64**	34.94	34.13	34.94	34.98	33.71
	MAPE(%)	8.85	8.95	8.88	8.73	8.68	8.88	8.622	8.88	8.85	**8.63**
	MNLL	–	–	3.80	9.88	**3.60**	3.78	22.31	3.80	–	**3.60**
	PICP(%)	–	91.83	93.86	53.74	**95.80**	**95.53**	38.05	94.78	94.65	**95.03**
	MPIW	–	96.63	112.99	31.16	112.36	127.00	19.03	118.79	118.83	**106.65**
PEMS08	MAE	15.95	16.40	16.29	15.87	15.51	16.29	15.79	16.29	15.95	**15.44**
	RMSE	25.22	25.79	25.71	25.05	24.64	25.71	24.96	25.71	25.22	**24.60**
	MAPE(%)	10.09	10.56	10.36	**10.05**	10.14	10.36	10.17	10.36	10.09	10.06
	MNLL	–	–	3.63	11.77	3.45	3.62	11.58	3.63	–	**3.44**
	PICP(%)	–	93.95	94.79	49.91	**95.88**	**97.15**	50.15	**95.37**	**95.16**	**95.35**
	MPIW	–	82.13	93.13	23.53	91.45	113.34	23.57	96.94	96.34	**87.90**

The results are evaluated according to the following criteria. Any PICP ≥ 95% is the best and the smallest corresponding MPIW is the best. If all the PICP < 95%, then the largest PICP is the best. MPIW only assessed when the corresponding PICP ≥ 95%. "-"means void values, i.e., the corresponding method cannot estimate uncertainty or likelihoods.



(a) PEMS03 (b) PEMS04

(c) PEMS07 (d) PEMS08

Fig. 10. Uncertainty quantification results on randomly selected road segments from different datasets.

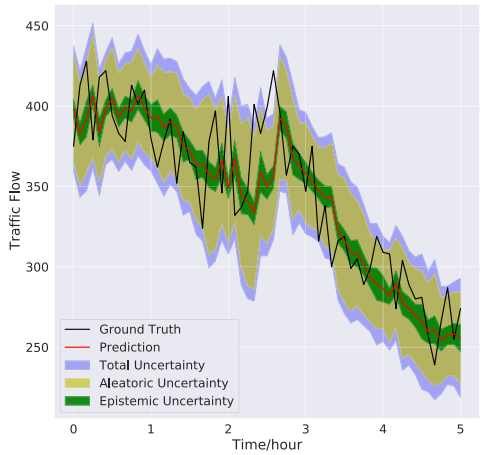
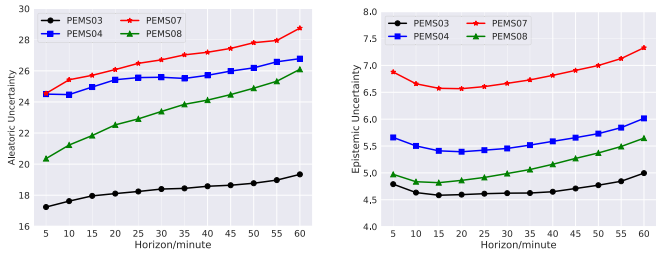


Fig. 11. Quantification results of different uncertainties on partial data from a randomly selected segment of PEMS08.



(a) Aleatoric uncertainty (b) Epistemic uncertainty

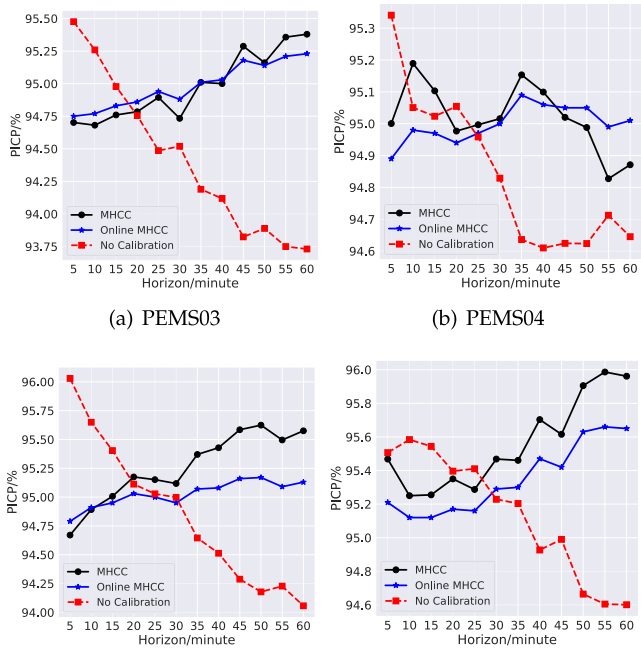
Fig. 12. Uncertainty quantification results with respect to different horizons.

According to the experimental results, we can also see that when only the epistemic uncertainty is considered using variational inference (MCDO) or deep ensembling (FGE), the traffic flow point prediction performance is improved compared to deterministic methods but the uncertainty quantification performance is poor. If merely the aleatoric uncertainty is taken into account (MVE, TS, Conformal, and CFRNN), the uncertainty quantification performance is satisfying while the point prediction slightly decreases compared to deterministic methods. On the other hand, if both the epistemic and aleatoric uncertainties are estimated, e.g., Combined and DeepSTUQ, the point prediction and uncertainty quantification performance are both improved.

G. Model Calibration Results

We compare the proposed calibration approaches, MHCC and online MHCC, to Split Conformal Prediction (SCP) [28], Local Weighted Conformal Inference (LWCI) [28] and TS. The results demonstrated in Table IV imply that online MHCC has the best overall (marginal) uncertainty quantification calibration performance.

In addition, we propose a new metric for evaluating the horizon-wise (conditional) uncertainty quantification performance, which is called Mean Horizon-wise Prediction Interval



(a) PEMS03 (b) PEMS04

(c) PEMS07 (d) PEMS08

Fig. 13. PICPs with respect to various forecast horizons, where solid and dashed lines denote MHCC and the uncalibrated model, respectively.

Coverage Error (MHPICE). Let e^h be the horizon-wise prediction interval coverage error at horizon h, then MHPICE can be expressed as follows:

$$e^h = \begin{cases} 0, & \text{if } p^h \geq 1-\alpha \\ 1-\alpha-p^h, & \text{else.} \end{cases} \tag{36}$$

Accordingly, MHPICE is defined as follows:

$$\text{MHPICE} = \frac{1}{\tau}\sum_{h=1}^{\tau} e^h. \tag{37}$$

Naturally, lower MHPICE means better performance.

From the results demonstrated in Fig. 13 and Table V, it can be seen that before MHCC, the test PICPs decrease as the horizon increase, which makes the uncertainty quantification results less reliable. After MHCC, the test PICPs at each horizon are closer to the target significance level, 95%, than before. This implies that MHCC can improve the credibility of the horizon-wise uncertainty quantification results.

H. Robustness Test

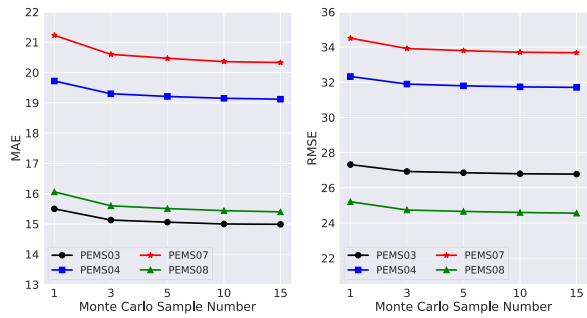
We report the MAE, RMSE, MAPE, MNLL, PICP, and MPIW results of the methods under Gaussian and non-Gaussian noises on PEMS04 to evaluate their robustness. From the results illustrated in Table VI, it can be seen that the proposed method are more resilient to noise in terms of point prediction performance. Besides, the online MHCC method has the best uncertainty quantification performance compared to other calibration methods.

TABLE IV
MODEL CALIBRATION RESULTS ON PEMS03, PEMS04, PEMS07, AND PEMS08

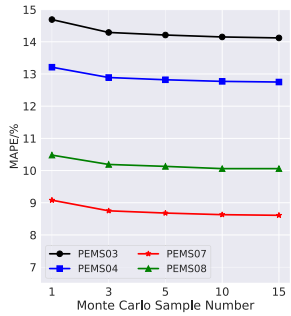
Dataset	Metrics	No Calibration	TS	SCP	LWCI	MHCC	Online MHCC
PEMS03	MNLL	3.39	**3.38**	-	-	-	-
	PICP(%)	94.22	94.75	93.25	94.27	94.97	**94.98**
	MPIW	74.51	76.91	88.02	75.89	79.58	79.86
PEMS04	MNLL	**3.57**	**3.57**	-	-	-	-
	PICP(%)	94.90	**95.23**	94.72	**95.09**	**95.02**	**95.00**
	MPIW	103.35	105.42	127.04	104.38	103.91	**103.76**
PEMS07	MNLL	**3.60**	**3.60**	-	-	-	-
	PICP(%)	**95.38**	**95.74**	94.27	**95.61**	**95.25**	**95.03**
	MPIW	108.85	111.68	127.90	111.81	108.37	**106.65**
PEMS08	MNLL	3.45	**3.44**	-	-	-	-
	PICP(%)	**96.28**	**95.65**	**95.00**	**95.94**	**95.55**	**95.35**
	MPIW	94.25	89.63	100.34	91.76	89.23	**87.90**

TABLE V
MHPICE ON PEMS03, PEMS04, PEMS07, AND PEMS08

Dataset	No Calibration	MHCC	Online MHCC
PEMS03	0.646	0.120	**0.080**
PEMS04	0.196	0.028	**0.021**
PEMS07	0.341	0.036	**0.033**
PEMS08	0.101	**0.000**	**0.000**



(a) MAE (b) RMSE



(c) MAPE

Fig. 14. Prediction results with respect to different numbers of Monte Carlo samples.

I. Ablation Study

Three groups of experiments are conducted to verify the effects of the proposed AWA training, the proposed calibration method, and different numbers of Monte Carlo samples, respectively.

1) Effect of AWA Re-Training: The prediction performance of the same pre-trained model prior to and following AWA post-processing re-training are compared. Table VII demonstrates that after AWA re-training, the point prediction performance has improved, indicating that the proposed AWA re-training method can approximate the deep ensembling method using only one single model with mere 20 additional epochs. The results suggest that the proposed AWA retraining method can also improve the performances of other methods, i.e., Point, MVE, and MCDO. Therefore, compared to conventional deep ensembling, DeepSTUQ requires less time and memory.

2) Effect of Monte Carlo Sample Number: To investigate how the number of Monte Carlo samples affects the model performance, the sample number is set to 1, 3, 5, 10 and 15. As shown in Fig. 14, the performance of the proposed method enhances as the number of Monte Carlo samples rises, and only a small number of Monte Carlo samples are required to provide high prediction performance. The performance gradually saturates when the sample size approaches 15. Accordingly, for the trade-off between the model performance and the inference time cost, the testing sample number can be fixed to 10 at test time.

J. Memory Cost and Computation Time

The quantitative results of the memory cost and computation time on PEMS08 are reported in Table VIII. From the results, we can be see that DeepSTUQ has almost same model sizes and training time with the Point prediction model, which is significantly efficient than the standard Deep Ensembles. The inference time and memory cost of DeepSTUQ are lightly larger than standard Deep Ensembles, but the inference time per step is less than 7.80 ms. Therefore, DeepSTUQ is scalable for large traffic forecasting datasets and applicable for the potential practical applications.

TABLE VI
ROBUSTNESS TEST RESULTS WITH DIFFERENT PERTURBATIONS ON PEMS4

Noise	Metrics	Point	MVE	No Calibration	MHCC	Online MHCC
Gaussian(0,10)	MAE	20.43	20.05	**19.18**	**19.18**	**19.18**
	RMSE	33.01	32.40	**31.75**	**31.75**	**31.75**
	MAPE(%)	13.78	13.61	**12.87**	**12.87**	**12.87**
	MNLL	–	3.73	**3.58**	–	–
	PICP(%)	–	92.9	**95.06**	**95.18**	**95.02**
	MPIW	–	102.99	103.98	104.89	**103.72**
Gaussian(0,20)	MAE	20.82	20.52	**19.31**	**19.31**	**19.31**
	RMSE	33.30	32.71	**31.81**	**31.81**	**31.81**
	MAPE(%)	14.52	14.50	**13.23**	**13.23**	**13.23**
	MNLL	–	3.78	**3.60**	–	–
	PICP(%)	–	92.21	**95.43**	**95.44**	**95.17**
	MPIW	–	103.02	106.65	106.79	104.72
Log-Gaussian(2,1)	MAE	23.23	22.38	**21.62**	**21.62**	**21.62**
	RMSE	35.48	34.18	**33.70**	**33.70**	**33.70**
	MAPE(%)	18.97	19.30	**18.28**	**18.28**	**18.28**
	MNLL	–	3.91	**3.73**	–	–
	PICP(%)	–	90.41	94.04	94.39	**94.49**
	MPIW	–	105.96	108.09	104.98	111.14

TABLE VII
ABLATION STUDY RESULTS ON AWA RE-TRAINING

Dataset	Metrics	Point		MVE		MCDO		DeepSTUQ	
		No AWA	AWA	No AWA	AWA	No AWA	AWA	No AWA	AWA
PEMS03	MAE	15.83	15.68	15.79	15.66	15.33	15.37	15.29	**15.13**
	RMSE	27.94	27.91	27.98	27.86	27.30	27.50	27.13	**26.77**
	MAPE(%)	14.79	14.66	14.71	14.52	14.29	14.25	14.60	**14.03**
PEMS04	MAE	19.98	19.96	19.74	19.66	19.24	19.12	19.23	**19.11**
	RMSE	32.64	32.72	31.99	32.04	31.66	31.59	31.73	**31.68**
	MAPE(%)	13.07	13.04	13.04	12.99	12.85	12.76	12.87	**12.71**
PEMS08	MAE	16.45	16.32	16.17	15.98	15.93	15.83	15.51	**15.44**
	RMSE	25.90	25.88	25.43	25.34	25.15	25.05	24.64	**24.60**
	MAPE(%)	10.59	10.37	10.70	10.20	10.28	10.10	10.14	**10.06**

TABLE VIII
MEMORY COST AND COMPUTATION TIME ON PEMS08 (CPU: AMD EPYC 7302, GPU: NVIDIA TESLA T4)

	Point	Deep Ensembles	DeepSTUQ
Model size (MB)	0.57	5.73	0.75
Training time (s/epoch)	25.46	254.63	29.67
Total inference time (s)	3.30	27.30	33.48
Memory cost (MB)	475.63	1,063.37	1,165.78

VI. CONCLUSION

In this paper, we introduce a novel and unified uncertainty quantification method for traffic forecasting called DeepSTUQ. The proposed method consists of three components. In the first component, to model the aleatoric uncertainty, a hybrid loss function is used to train a base spatio-temporal model. The second component aims to model the epistemic uncertainty, where the merits of variational inference and deep ensembling are combined through the dropout pre-training and AWA re-training. Finally, the model is calibrated on the validation dataset using a post-processing calibration method based on Temperature Scaling to further improve the uncertainty estimation performance. Four distinct public datasets are then subjected to thorough experiments. The results indicate that DeepSTUQ outperforms contemporary state-of-the-art spatio-temporal models and uncertainty quantification methods. Moreover, the proposed DeepSTUQ can achieve high robustness against noise.

In terms of future work, we plan to explore other techniques to design novel model architectures. Another possible direction is to study data interpolation and imputation in traffic forecasting.

ACKNOWLEDGMENTS

The authors would like to thank James J.Q. Yu for his insightful suggestions.

REFERENCES

[1] M. Abdar et al., "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Inf. Fusion*, vol. 76, pp. 243–297, 2021.
[2] W. Alajali, W. Zhou, and S. Wen, "Traffic flow prediction for road intersection safety," in *Proc. IEEE SmartWorld Ubiquitous Intell. Comput. Adv. Trusted Comput. Scalable Comput. Commun. Cloud Big Data Comput. Internet People Smart City Innov.*, 2018, pp. 812–820.
[3] P. Alquier, "User-friendly introduction to PAC-Bayes bounds," 2021, *arXiv:2110.11216*.
[4] A. N. Angelopoulos and S. Bates, "A gentle introduction to conformal prediction and distribution-free uncertainty quantification," 2021, *arXiv:2107.07511*.

[5] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 17804–17815.

[6] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1613–1622.

[7] K. Cho et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*.

[8] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 933–941.

[9] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 3844–3852.

[10] S. Fort, H. Hu, and B. Lakshminarayanan, "Deep ensembles: A loss landscape perspective," 2019, *arXiv: 1912.02757*.

[11] T. Fu et al., "Traffic safety oriented multi-intersection flow prediction based on transformer and CNN," *Secur. Commun. Netw.*, vol. 2023, pp. 1–13, 2023.

[12] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1050–1059.

[13] T. Garipov, P. Izmailov, D. Podoprikhin, D. P. Vetrov, and A. G. Wilson, "Loss surfaces, mode connectivity, and fast ensembling of DNNs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 8803–8812.

[14] J. Gawlikowski et al., "A survey of uncertainty in deep neural networks," 2021, *arXiv:2107.03342*.

[15] P. Germain, F. Bach, A. Lacoste, and S. Lacoste-Julien, "PAC-Bayesian theory meets Bayesian inference," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 1876–1884.

[16] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1321–1330.

[17] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 922–929.

[18] W. Hoeffding, "Probability inequalities for sums of bounded random variables," in *The Collected Works of Wassily Hoeffding*. Berlin, Germany: Springer, 1994, pp. 409–426.

[19] P. Izmailov, A. Wilson, D. Podoprikhin, D. Vetrov, and T. Garipov, "Averaging weights leads to wider optima and better generalization," in *Proc. 34th Conf. Uncertainty Artif. Intell.*, 2018, pp. 876–885.

[20] W. Jiang and J. Luo, "Graph neural network for traffic forecasting: A survey," 2021, *arXiv:2101.11174*.

[21] L. V. Jospin, W. Buntine, F. Boussaid, H. Laga, and M. Bennamoun, "Hands-on Bayesian neural networks–A tutorial for deep learning users," 2020, *arXiv: 2007.06823*.

[22] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?," 2017, *arXiv: 1703.04977*.

[23] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," 2016, *arXiv:1609.04836*.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[25] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.

[26] R. Koenker and K. F. Hallock, "Quantile regression," *J. Econ. Perspectives*, vol. 15, no. 4, pp. 143–156, 2001.

[27] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6405–6416.

[28] J. Lei, M. G'Sell, A. Rinaldo, R. J. Tibshirani, and L. Wasserman, "Distribution-free predictive inference for regression," *J. Amer. Statist. Assoc.*, vol. 113, no. 523, pp. 1094–1111, 2018.

[29] M. Li and Z. Zhu, "Spatial-temporal fusion graph neural networks for traffic flow forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 4189–4196.

[30] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–16.

[31] Y. Liang et al., "Mixed-order relation-aware recurrent neural networks for spatio-temporal forecasting," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 9, pp. 9254–9268, Sep. 2023.

[32] D. A. McAllester, "PAC-Bayesian model averaging," in *Proc. 12th Annu. Conf. Comput. Learn. Theory*, 1999, pp. 164–170.

[33] H. Miao, J. Shen, J. Cao, J. Xia, and S. Wang, "MBA-STNet: Bayes-enhanced discriminative multi-task learning for flow prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 7, pp. 7164–7177, Jul. 2023.

[34] S. M. Mousavi, O. A. Osman, D. Lord, K. K. Dixon, and B. Dadashova, "Investigating the safety and operational benefits of mixed traffic environments with different automated vehicle market penetration rates in the proximity of a driveway on an urban arterial," *Accident Anal. Prevention*, vol. 152, 2021, Art. no. 105982.

[35] D. A. Nix and A. S. Weigend, "Estimating the mean and variance of the target probability distribution," in *Proc. IEEE Int. Conf. Neural Netw.*, 1994, pp. 55–60.

[36] W. Qian, D. Zhang, Y. Zhao, K. Zheng, and J. James, "Uncertainty quantification for traffic forecasting: A unified approach," in *Proc. IEEE 39th Int. Conf. Data Eng.*, 2023, pp. 992–1004.

[37] H. Qu, Y. Gong, M. Chen, J. Zhang, Y. Zheng, and Y. Yin, "Forecasting fine-grained urban flows via spatio-temporal contrastive self-supervision," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 8, pp. 8008–8023, Aug. 2023.

[38] A. Sohail, M. A. Cheema, M. E. Ali, A. N. Toosi, and H. A. Rakha, "Data-driven approaches for road safety: A comprehensive systematic literature review," *Saf. Sci.*, vol. 158, 2023, Art. no. 105949.

[39] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 914–921.

[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[41] K. Stankeviciute, A. M. Alaa, and M. van der Schaar, "Conformal time-series forecasting," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 6216–6228.

[42] A. Van Den Oord et al., "WaveNet: A generative model for raw audio," in *Proc. 9th ISCA Speech Synth. Workshop*, 2016, Art. no. 125.

[43] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–12.

[44] M. Veres and M. Moussa, "Deep learning for intelligent transportation systems: A survey of emerging trends," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3152–3168, Aug. 2020.

[45] B. Wang, T. Li, Z. Yan, G. Zhang, and J. Lu, "DeepPIPE: A distribution-free uncertainty quantification approach for time series forecasting," *Neurocomputing*, vol. 397, pp. 11–19, 2020.

[46] B. Wang et al., "Deep uncertainty quantification: A machine learning approach for weather forecasting," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 2087–2095.

[47] S. Wang, J. Cao, and P. Yu, "Deep learning for spatio-temporal data mining: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 8, pp. 3681–3700, Aug. 2022.

[48] Z. Wang et al., "Forecasting ambulance demand with profiled human mobility via heterogeneous multi-graph neural networks," in *Proc. IEEE 37th Int. Conf. Data Eng.*, 2021, pp. 1751–1762.

[49] A. G. Wilson and P. Izmailov, "Bayesian deep learning and a probabilistic perspective of generalization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 4697–4708.

[50] D. Wu et al., "Quantifying uncertainty in deep spatiotemporal forecasting," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 1841–1851.

[51] 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 *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 753–763.

[52] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 1907–1913.

[53] Z. Xie, X. Wang, H. Zhang, I. Sato, and M. Sugiyama, "Adaptive inertia: Disentangling the effects of adaptive learning rate and momentum," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 24430–24459.

[54] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 3634–3640.

[55] X. Zhang et al., "Traffic flow forecasting with spatial-temporal graph diffusion network," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 15008–15015.

[56] L. Zhao et al., "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 9, pp. 3848–3858, Sep. 2020.

[57] C. Zheng, X. Fan, C. Wang, and J. Qi, "GMAN: A graph multi-attention network for traffic prediction," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 1234–1241.
[58] Z. Zhou, Y. Wang, X. Xie, L. Qiao, and Y. Li, "STUaNet: Understanding uncertainty in spatiotemporal collective human mobility," in *Proc. Web Conf.*, 2021, pp. 1868–1879.
[59] L. Zhu and N. Laptev, "Deep and confident prediction for time series at Uber," in *Proc. IEEE Int. Conf. Data Mining Workshops*, 2017, pp. 103–110.
[60] D. Zhuang, S. Wang, H. Koutsopoulos, and J. Zhao, "Uncertainty quantification of sparse travel demand prediction with spatial-temporal graph neural networks," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 4639–4647.

Weizhu Qian received the PhD degree in computer science from the Université Bourgogne Franche-Comté, in 2021. He is a lecturer with Soochow University, China. He was a postdoctoral researcher with Aalborg University. His research interests mainly focus on deep learning and data science.

Yan Zhao received the doctoral degree in computer science from Soochow University, in 2020. She is an associate professor with Aalborg University. Her research interests include spatial database and trajectory computing.

Dalin Zhang (Member, IEEE) received the doctoral degree in computer science from the UNSW Sydney, in 2020. He is currently an assistant professor with Aalborg University, Denmark. His research interests include spatiotemporal data mining, healthe informatics, and brain-computer interface.

Bowei Chen received the PhD degree in computer science from the University College London. He is an associate professor in marketing analytics and data science with the Adam Smith Business School, University of Glasgow. He has broad research interests related to the applications of probabilistic modeling and deep learning in business analytics. His research has appeared in the *IEEE Transactions on Knowledge and Data Engineering*, *ACM Transactions on Intelligent Systems and Technology*, *IEEE Transactions on Network Science and Engineering*, *European Journal of Operational Research*, SIGKDD, SIGIR, and AAAI.

Kai Zheng (Senior Member, IEEE) received the PhD degree in computer science from the University of Queensland, in 2012. He is a professor of computer science with the University of Electronic Science and Technology of China. He has been working in the area of spatial-temporal databases, uncertain databases, social-media analysis, in-memory computing, and blockchain technologies. He has published more than 100 papers in prestigious journals and conferences in data management field such as SIGMOD, ICDE, the *VLDB Journal*, ACM Transactions and IEEE Transactions.

Xiaofang Zhou (Fellow, IEEE) received the bachelor's and master's degrees in computer science from Nanjing University, in 1984 and 1987, respectively, and the PhD degree in computer science from the University of Queensland, in 1994. He is the Otto Poon professor of engineering and chair professor of computer science and engineering with the Hong Kong University of Science and Technology. He is head of the Department of Computer Science and Engineering. His research is focused on finding effective and efficient solutions to managing integrating, and analyzing very large amounts of complex data for business and scientific applications. His research interests include spatial and multimedia databases, high performance query processing, web information systems, data mining, and data quality management.