Adaptive Modeling of Uncertainties for Traffic Forecasting

Ying Wu, Yongchao Ye, *Student Member, IEEE*, Adnan Zeb, James Jianqiao Yu, *Senior Member, IEEE*, and Zheng Wang

***Abstract*— Deep neural networks (DNNs) have emerged as a dominant approach for developing traffic forecasting models. These models are typically trained to minimize error on averaged test cases and produce a single-point prediction, such as a scalar value for traffic speed or travel time. However, single-point predictions fail to account for prediction uncertainty that is critical for many transportation management scenarios, such as determining the best- or worst-case arrival time. We present QUANTRAFFIC, a generic framework to enhance the capability of an arbitrary DNN model for uncertainty modeling. QUANTRAFFIC requires little human involvement and does not change the base DNN architecture during deployment. Instead, it automatically learns a standard quantile function during the DNN model training to produce a prediction interval for the single-point prediction. The prediction interval defines a range where the true value of the traffic prediction is likely to fall. Furthermore, QUANTRAFFIC develops an adaptive scheme that dynamically adjusts the prediction interval based on the location and prediction window of the test input. We evaluated QUANTRAFFIC by applying it to five representative DNN models for traffic forecasting across seven public datasets. We then compared QUANTRAFFIC against six uncertainty quantification methods. Compared to the baseline uncertainty modeling techniques, QUANTRAFFIC with base DNN architectures delivers consistently better and more robust performance than the existing ones on the reported datasets.**

***Index Terms*— Traffic prediction, uncertainty qualification, quantile model.**

I. INTRODUCTION

ACCURATE prediction of future traffic information, including traffic volume, congestion levels, traffic speed, and travel time, is crucial for a variety of transportation applications, such as congestion control [1], travel time estimation [2], emergent route planning [3], and taxi demand prediction [4]. It also supports effective transportation planning and management, enabling policymakers to optimize traffic flow and emergency response planning while providing road users with a safer and more efficient travel experience.

In recent years, there has been a growing interest in the development of advanced predictive models for traffic information using data-driven approaches such as DNNs [5]. Multiple studies have shown that traffic forecasting models based on DNNs outperform classical machine learning methods by a large margin [6], [7]. However, these models are typically trained to minimize the averaged prediction error, resulting in a considerable variation in performance across test samples [8]. This variation poses significant challenges for individual use cases where precise traffic prediction is critical but difficult to achieve. For instance, accurately predicting traffic at crossroads with high traffic volumes during rush hours is crucial for urban travel but can be challenging due to complex traffic patterns [9]. As the accuracy of traffic forecasting can fluctuate over time [10], it is crucial to model and quantify the prediction uncertainty of the model for individual roads, locations, and sensors for a given prediction window (e.g., traffic predictions for the next n minutes).

Modeling and quantifying traffic forecasting uncertainties have real-world use cases in transportation management. For example, knowing the upper bound of a traffic flow prediction in emergency response situations can help emergency responders avoid congested roads and take the quickest and safest route to their destination. Similarly, knowing the earliest and worst-case travel times enables users to make informed decisions about their travel plans, such as selecting the most convenient transportation method while minimizing the likelihood of being late to an appointment or arriving too early. Therefore, by quantifying the forecasting uncertainty, we can improve the reliability of traffic prediction and traffic management efficiency. Unfortunately, despite the huge benefits of uncertainty modeling, prior research on traffic forecasting has largely overlooked this issue. This is a massively missed opportunity.

While uncertainty quantification has gained popularity in several deep learning domains, such as computer vision [11], human mobility [12], and medical applications [13], it is noteworthy that the modeling uncertainties in traffic forecasting has only recently started to receive attention. Recent studies, such as those presented in [14] and [15], represent some of the

Manuscript received 12 March 2023; revised 20 July 2023 and 23 September 2023; accepted 9 October 2023. This work was supported in part by a joint Ph.D. Program between the Southern University of Science and Technology and the University of Leeds. The Associate Editor for this article was C. K. Ahn. *(Corresponding authors: James Jianqiao Yu; Zheng Wang.)*

Ying Wu is with the School of Computing, University of Leeds, LS2 9JT Leeds, U.K., and also with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China.

Yongchao Ye and Adnan Zeb are with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China.

James Jianqiao Yu is with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China, and also with the Department of Computer Science, University of York, YO10 5DD York, U.K. (e-mail: yujq3@sustech.edu.cn).

Zheng Wang is with the School of Computing, University of Leeds, LS2 9JT Leeds, U.K. (e-mail: z.wang5@leeds.ac.uk).

Digital Object Identifier 10.1109/TITS.2023.3327100

initial attempts in this direction. Reference [14] uses a classical statistical method to quantify the uncertainties associated with average daily traffic volume forecasts. However, this approach requires manual tuning and selecting a set of features for each dataset to fit a linear model. Its requirement of intensive expert involvement thus limits its practicability. In [15], DeepSTUQ is employed to estimate the data and model uncertainties in traffic forecasting, based on variational inference and deep ensemble learning. This approach provides a more comprehensive understanding of the uncertainties involved, but it requires significant changes to the original model architectures, which limits its generalization ability.

In this paper, we present QUANTRAFFIC, a generic framework for quantifying the prediction uncertainties of a DNN-based traffic forecasting model. Unlike prior work [14], [15], QUANTRAFFIC is designed to minimize engineering efforts and expert involvement. It can work with any DNN model without changing the underlying architecture during deployment. By producing a prediction interval (**PI**) that captures the range in which the true value (such as travel time) is likely to fall, QUANTRAFFIC enhances the capability of a standard DNN to capture prediction uncertainties.

At the core of QUANTRAFFIC is a quantile function built upon the recently proposed Conformalized Quantile Regression (**CQR**) algorithm [16]. The quantile function estimates the PI of a given model output based on the data distribution and validation errors observed during the standard DNN model training process. QUANTRAFFIC is designed to simplify the training and usage of the quantile function. The process involves attaching a linear layer to the last layer of the base DNN model and using a pinball loss function [17] during standard DNN training. Once trained, the base DNN model and the quantile function can be used as standalone components during deployment. During inference, the quantile function generates an initial PI based on the DNN model's single-point prediction. Then, an adjustment is made to refine it and improve its accuracy. The goal is to increase the coverage of the PI while simultaneously minimizing the width between its upper and lower bounds.

Unlike standard CQR, which uses a constant global value to adjust the initial PI, QUANTRAFFIC develops an adaptive scheme to tailor the adjustment value applied to the initial PI for the specific location (or sensor node) of the test sample within a given prediction window. This allows the uncertainty method to consider the prediction difficulty of each test sample. For example, locations with high variability in traffic patterns may require a larger adjustment to achieve accurate PIs. In contrast, those with more predictable traffic patterns may require a smaller value. We achieve this by utilizing a calibration table that is automatically constructed using an optimization function on a calibration dataset. This table provides the optimal adjustment for a node-prediction-window combination, enabling QUANTRAFFIC to account for unique prediction challenges in each test sample. By using differentiated residues to adjust the initial PI, we achieve greater precision and reliability over the standard CQR.

We have implemented a working prototype of QUANTRAFFIC, which will be open-sourced upon acceptance of this work. We evaluate QUANTRAFFIC by applying it to five representative DNN architectures [18], [19], [20], [21], [22] for traffic forecasting. We then test the QUANTRAFFIC-enhanced DNN model on seven public datasets for traffic speed and flow prediction. We compare QUANTRAFFIC against six state-of-the-art uncertainty modeling methods [23], [24], [25], [26], [27] and two classical methods based on historical data. Experimental results show that QUANTRAFFIC consistently outperforms competing baselines across DNN models and datasets, delivering better and more robust performance for uncertainty quantification.

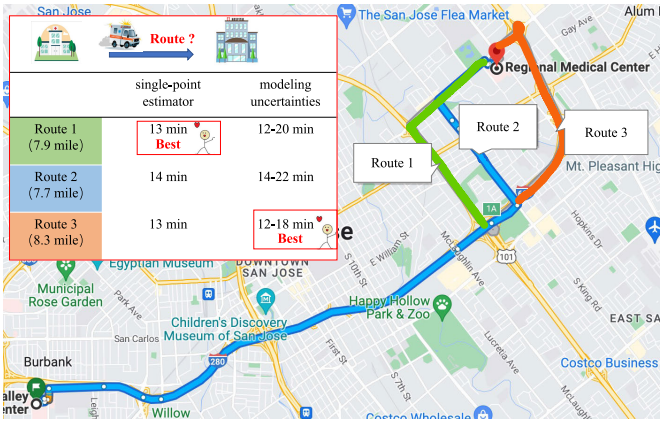


Fig. 1. **A representative emergency route planning example between two points.** A single-point travel time estimation does not provide the upper-bound travel time for route selection which is important for ensuring the worst-case arrival time.

This paper makes the following contributions:

- It presents a generic framework for modeling the uncertainty of predictions in DNN-based traffic forecasting models without requiring any modifications to the underlying DNN prediction model (Section V);
- It develops an adaptive scheme to consider the prediction variances of individual locations, leading to more robust results than standard uncertainty modeling methods (Section V-D);
- It provides a large independent study to highlight the importance of uncertainty modeling of traffic forecasting. We hope our study can encourage further research along this line (Section VII).

A. Data Availability

The data and code associated with this paper are openly available at: https://github.com/wuyingvia/QUANTRAFFIC

II. MOTIVATION

As a motivative example, consider the emergency route planning task depicted in Figure 1. This example represents a day-to-day scenario where traffic forecasts are employed to plan optimal routes, like an ambulance traveling from Regional Medical Center to Santa Clara Valley Medical Center. The objective is to identify the quickest route based on the predicted traffic while ensuring the worst-case arrival time.

Typical traffic forecasting models can provide estimates for travel time of different routes, as seen in Figure 1. However, relying solely on these single-point forecasts for route planning

can lead to unreliable decisions since they lack information about the uncertainty or confidence of the predictions. In the example shown, a user may choose Route 1 due to its lowest average travel time (13 minutes). Nevertheless, this decision overlooks the potential variability in traffic dynamics and route complexity, possibly rendering a worse real travel time. To account for this variability and provide worst-case scenario estimates, it is essential to have confidence intervals associated with the travel time predictions. Incorporating such confidence estimation can have a significant impact on time-critical route planning tasks, a factor that is overlooked in the current literature.

We show how estimating the uncertainty of travel time predictions can provide valuable information for time-critical route planning. By providing travel time bounds besides single-point forecasts, we can tell that selected routes have much higher variability in their predicted travel times than others. If the uncertainty estimation is accurate, the additional information help users avoid routes that are more uncertain on travel times, even if their average predicted travel time is lower. For example, in the scenario depicted in Figure 1, one may finally choose Route 3 despite it being the longest, in order to ensure the worst-case arrival time.

This example demonstrates the limitations of single-point-based traffic forecasting models for time-critical travel planning. This highlights the need for uncertainty modeling in traffic forecasting. To address this, our work proposes a generic framework for precise single-point and bound estimations to better model traffic prediction uncertainties.

III. Background and Related Work

Our work builds upon the following past foundations, but our focus differs from each.

A. Spatio-Temporal Traffic Forecasting

Traffic forecasting is a well-established research topic with a wide range of proposed solutions. Classical statistical methods, such as historical average, regression, and integrated moving average models, have been explored in the past [28]. However, more recent research has leveraged DNNs to model the spatio-temporal correlations in traffic data. Compared to classical statistical methods, DNNs can better capture complex relationships in historical data while avoiding the need for hand-engineered features. Researchers have explored several approaches to represent traffic data, including temporal sequence modeling with recurrent neural networks [29], multidimensional matrix representations with convolutional neural networks [30], and graph neural networks (GNN) [7]. DNN-based methods have been shown to deliver state-of-the-art results in various traffic-related tasks, such as ride-sharing [31], and travel planning [32]. Due to the better performance over alternative methods, DNNs have emerged as the dominant approach for building traffic forecasting models.

The majority of existing DNN-based traffic forecasting models only provide a single-point prediction such as the travel time. However, a single-point estimation only reflects the average traffic scenario but not the best or worse cases. As highlighted in Section II, the upper and lower bounds of the travel time can be critical for choosing the best route in time-critical route planning tasks. This requires one to consider the uncertainty of the predicted travel time and to produce metrics similar to statistical confidence intervals. This work aims to address this issue by developing a generic approach that can provide such information from any DNN model, making it applicable to a wide range of traffic forecasting architectures.

B. Modeling Prediction Uncertainty

Although uncertainty modeling has been largely overlooked in prior traffic forecasting approaches, it has drawn much attention in other DNN-based modeling tasks. Various techniques have been proposed to quantify prediction errors, confidences, or uncertainties. These methods can be broadly categorized as Bayesian and Frequentist ones, which have been extensively studied in the literature [33], [34].

1) Bayesian Models: provide a robust probabilistic framework for modeling uncertainty with Bayesian statistics [35]. In this approach, the model incorporates prior knowledge or beliefs for parameter initialization and infers the posterior distribution using the likelihood function between the data and a predefined initial distribution. Techniques for quantifying uncertainty in DNNs using Bayesian models include Monte Carlo (MC) dropout [25] and Variational Inference [24]. However, there are emerging challenges, such as relatively low computation efficiency and strong prior distribution assumptions. These challenges can be particularly acute in high-dimensional models or large datasets.

2) Frequentist Methods: provide predictions based on a single forward pass with a deterministic network and quantify uncertainty by using additional qualification schemes. These methods use post-hoc calibrations, such as conformal prediction and differentiable modeling structures, and their loss objectives, such as quantile prediction [36], to capture uncertainties. Ensemble methods, such as those that use random initialization or a mixture of experts [37], retrain models on partial datasets, or adopt data augmentation techniques (e.g., cross-validation [38], and bootstrap aggregating [39]), are also considered frequentist methods. However, ensemble methods require trial-and-error adjustments to parameters without a solid mathematical foundation, leading to poor coverage guarantees. Moreover, frequentist methods are often overconfident [36], [40], which can result in inaccurate uncertainty estimates. Despite these limitations, frequentist methods are attractive because they are computationally efficient and do not require prior assumptions on the model or data distribution.

C. Summary

In summary, existing methods for quantifying uncertainty suffer from several issues, including inaccurate coverage guarantees, strong distributional assumptions, and insufficiently calibrated prediction intervals. These challenges are further compounded when dealing with heteroscedastic data.[1]

[1]Heteroscedastic data refers to data with varying levels of variability or scatter across its range, as opposed to homoscedastic data, which has consistent levels of variability or scatter. An example of heteroscedastic data in traffic forecasting is rush hour traffic, which typically has more variability than traffic during off-peak times.

To address these issues, we propose an adaptive conformalized quantile model that provides a unified and reliable framework for quantifying uncertainty in traffic forecasting. Our approach is one of the first attempts to use frequentist methods for estimating uncertainty in traffic forecasting. We provide a comprehensive and structured comparison of existing approaches to real traffic data using various state-of-the-art DNN-based traffic forecasting models. We hope our work can promote more research in this important area of uncertainty quantification for traffic forecasting.

IV. Preliminaries

A. Problem Definition

A single-point traffic forecasting model attempts to predict future traffic information based on the past. Examples of traffic information include flow, speed, and density. Given a D-dimensional multivariate time series $X = \{x_{t-(m-1)}, \ldots, x_{t-1}, x_t\} \in \mathbb{R}^{N\times D}$ collected at t time with past m steps from N data sources (roads or sensors), a point forecasting model $f(\cdot)$ attempts to estimate the multivariate time series $Y = \{y_t, \ldots, y_{t+h-1}, y_{t+h}\} \in \mathbb{R}^{N\times D}$ in the next prediciton window (e.g., h steps), : $X \overset{f(\cdot)}{\rightarrow} Y$.

Probabilistic traffic forecasting involves predicting the likelihood of various potential outcomes rather than estimating the most likely outcome. Given historical data X, a probability predictor $F(\cdot)$ can estimate the uncertainty of future traffic conditions by producing a set of PIs, which contain the real data with a certain level of confidence.

Specifically, a probabilistic forecasting model attempts to estimate PIs for a set of future values denoted as $\hat{Y} = \{\hat{y}_t, \ldots, \hat{y}_{t+h-1}, \hat{y}_{t+h}\} \in \mathbb{R}^{2\times N\times D}$, where t is the current time, h is the prediction window. The PI for $\hat{y}_{t+h}$ is defined as $\hat{y}^l_{t+h} \leq \hat{y}_{t+h} \leq \hat{y}^u_{t+h}$, where $\hat{y}^l_{t+h}$ and $\hat{y}^u_{t+h}$ are the lower and upper bounds of the PI, respectively. By specifying a confidence level α, we can say that there is a probability of $(1-\alpha)$ that the true value Y_{t+h} falls within the PI of $\hat{Y}_{t+h}$. The probabilistic forecasting process can be described as $X \overset{F(\cdot)}{\rightarrow} \hat{Y}$, where $F(\cdot)$ is the probability predictor that maps historical data X to the predicted PIs for $\hat{Y}$.

This paper aims to develop a generic framework to extend a single-point predictor f to model the uncertainties in traffic modeling tasks. We target DNN-based traffic forecasting models as they are the core of state-of-the-art traffic forecasting methods [7]. It is worth noting that our approach does not change the underlying DNN model structure and can be integrated with other classical supervised learning methods, including support vector machines [41], linear [42] and nonlinear regressors [43].

B. Optimization Goals

Our goal is to generate PIs that are narrow and accurate, meaning that they are just wide enough to cover the true values (e.g., the actual traffic speed) with high probability. To achieve this, we need the PI to be *discriminative* so that it is narrow for single-point predictions with high confidence but wide enough to ensure good coverage overall [44].

We also require the uncertainty method to perform well across test samples and have good coverage, so we need the results to be *valid* [44]. Specifically, for a given probability or coverage rate of x (defined as $1-\alpha$), we expect the PI to encompass the true value at least x of the time.

C. Quantified Metrics

We adopt the established practices of prior uncertainty modeling methods in related domains [34], [44] and use two metrics to evaluate the efficacy of the PI: **validity** and **discrimination** [44]. Specifically, we use *mean prediction interval width (MPIW)* and *prediction interval coverage probability (PICP)* to measure these requirements. We aim is to obtain a small MPIW with adequate coverage across test samples. While a large MPIW can ensure good coverage, it may frequently underestimate or overestimate traffic information, even when the model is relatively confident with the prediction. On the other hand, a small MPIW can miss the true value, leading to incorrect decision such as selecting a slower route due to incorrect prediction.

1) Mean Prediction Interval Width: We compute the MPIW across n test samples as:

$$\text{MPIW} = \frac{1}{n}\sum_{i=1}^{n}\left|y^u_i - y^l_i\right|. \tag{1}$$

The MPIW is a *smaller-is-better* metric as we anticipate the prediction width as narrow as possible.

2) Prediction Interval Coverage Probability: In addition to MPIW, we also consider the frequency PI that covers the real value over the complete n test samples. The observed coverage $\hat{C}_\alpha$ is computed as:

$$\hat{C}_\alpha = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} c_i,\ c_i = \begin{cases} 1, & y_i \in [y^l_i, y^u_i] \\ 0, & y_i \notin [y^l_i, y^u_i] \end{cases}. \tag{2}$$

where c_i is a binary value counting if the true value y_i is within the PI $\{y^l_i, y^u_i\}$ for the test sample i.

Given a target probability level $1-\alpha$, the *expected coverage* is $1-\alpha$ for n test samples. α is defined as a mis-coverage rate. For this case, an ideal uncertainty predictor produces n PIs for all test samples to cover the real value of at least $1-\alpha$ time, rendering the *observed coverage* $\hat{C}_\alpha$, to be $1-\alpha$ as well. If $\hat{C}_\alpha$ is less than $1-\alpha$, the PI is likely undercovered.

Undercoverage is generally non-intuitive since it limits the PI to meet the requirement of retaining the true values of important test samples.

D. Conformalized Quantile Regression

Our approach is inspired by the recently proposed CQR algorithm [16]. CQR combines conformal prediction [23] and classical quantile regression [26] for uncertainty qualification. It uses quantile regression to generate an initial PI and then uses conformal prediction to adjust the PI if required.

1) Working Mechanism of CQR: In layperson's terms, a quantile regressor is a tool for quantifying the chance of having a specific point in a range of possible outcomes. For instance, if we want to predict the 90th percentile or quantile for traffic speed, there is a 5% chance that the actual speed

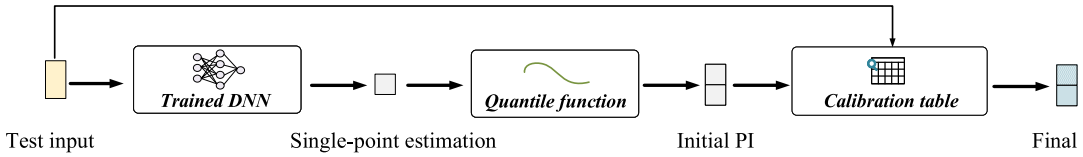


Fig. 2. **Modeling forecasting uncertainties during deployment.** QUANTRAFFIC extends the standard DNN-based traffic forecasting model to quantify prediction uncertainties. For a given test input, a trained DNN generates a single-point prediction akin to traditional DNN models. This single-point prediction is subsequently processed by a quantile function, which produces an initial PI for the input. The initial PI, along with the test input, is then passed through the calibration component to produce a final PI.

will be lower than our prediction and a 5% chance it will be higher. By fitting two quantile functions - one at the 5th percentile and another at the 95th percentile - we can create a PI with a 90% coverage. This is where CQR comes in handy for modeling traffic uncertainty, allowing us to estimate these quantiles. However, no previous work has used CQR for traffic uncertainty modeling, making our work the first attempt to fill this research gap.

In CQR, the training data is divided into two distinct sets: a training set and a calibration set. The training set is used to develop the traffic forecasting model using standard supervised learning techniques. Then, two quantile regressors are trained on the training set to generate initial estimates of the upper and lower bounds of the PI. One quantile regressor is used for estimating the upper bound quantile and the other for the lower bound. After obtaining the initial estimates, a conformal step is performed on the calibration to learn a *single* adjustment (e.g., increasing or decreasing the PI by x%) to be used to adjust the PI for all test samples.

2) Limitations of CQR: A drawback of using CQR for traffic forecasting is that it only produces a single, global quantile adjustment for all test samples. However, a global quantile adjustment is unlikely to be effective, as prediction difficulties can vary significantly across domains or roads. To illustrate this point, consider once again Routes 1 and 2 given in our motivation example (Section II). Suppose the traffic conditions on Route 1 are more complex than those on Route 2. Applying a global adjustment, such as increasing or decreasing the model prediction by a certain percentage, to both Routes 1 and 2 is unlikely to be effective. It could result in a PI that is too narrow for the complex Route 1 or too wide for the smooth Route 2. A better approach, which is adopted in this paper, is to generate different adjustment ranges for each route, resulting in more accurate and reliable PIs. Doing so enhances the robustness of traffic uncertainty modeling by considering the specific prediction difficulties of each location.

V. OUR APPROACH

QUANTRAFFIC is a generic framework for quantifying the prediction uncertainties of DNN-based traffic forecasting models. Given a certain confidence level, the QUANTRAFFIC framework computes a PI that defines a range of possible values where the real data (e.g., traffic time, flow or speed) is likely to fall within.

QUANTRAFFIC extends the existing CQR framework but offers a significant advantage by incorporating a dedicated calibration component. This component dynamically adjusts the width of the PI based on locations and prediction windows (e.g., traffic conditions in the next n minutes, also known as the *prediction horizon*), resulting in improved accuracy. Another important feature of QUANTRAFFIC is its ability to integrate without requiring modifications to the underlying DNN model structure, making it easy to implement. Instead, QUANTRAFFIC uses a dedicated loss function to train the DNN model and a quantile function to predict an initial PI. This seamless integration allows QUANTRAFFIC to efficiently and effectively enhance traffic forecasting in any DNN-based traffic predictor.

This section begins by providing an overview of QUANTRAFFIC and illustrating its design principle. It then discusses how to train and construct the key components that drive the DNN model to achieve better coverage and more accurate PIs. It also elaborates on how uncertainty qualification is achieved with test samples during deployment.

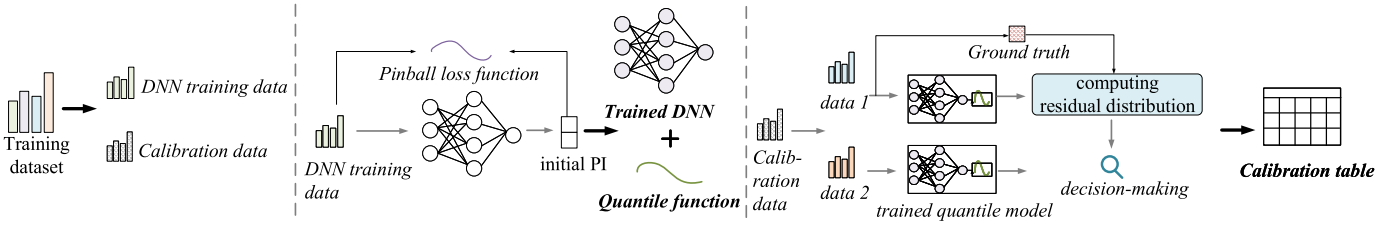
A. Overview of QUANTRAFFIC

Figure 2 demonstrates how QUANTRAFFIC enhances a standard DNN-based traffic forecasting model to quantify uncertainties for traffic forecasting during deployment. For a given test sample (e.g. traffic speed or flow), the trained DNN generates a single-point prediction that is then passed to the quantile function to create the initial PI. This PI is a 2-dimensional numerical vector with lower and upper bound values. Using the node id of the test sample and the prediction window, we search the calibration table for the corresponding residual value, which is used to amend the initial PI. Through this process, the PI can be more accurately adjusted to reflect the specific context of the prediction, leading to enhanced accuracy and better coverage.

In Figure 3, our approach adapts the standard DNN model training workflow for uncertainty modeling. Our training process starts by partitioning the training data into two *disjoint*, namely *training* and *calibration* sets, respectively, to train the DNN model and build the calibration table separately. This process is described in Section V-B. Specifically, the first training set is used to train both the DNN-based traffic forecasting model and a quantile function, using a pinball function to compute the loss during training (Section V-C). Once the DNN model and quantile function have been trained, we then use the calibration set to construct the calibration table (Section V-D).

B. Pre-Processing of Training Data

As shown in Figure 3(a), as a *one-off* preprocessing step, the training data is split into a standard model training dataset



(a) Pre-processing (b) Quantile DNN model training (c) Calibration table construction

Fig. 3. **Training workflow of our approach.** We first partition the DNN model training datasets into disjoint parts (a). We use the first part to train the DNN forecasting model and a quantile function that can produce an initial PI during deployment (b). We then apply the trained DNN model and quantile function to the set-aside calibration dataset to build the calibration component (c).

and a calibration dataset. Specifically, in this paper, we split the time series data of each node in chronological order. For calibration purposes, we allocate 40% of the training data, and this ratio can be adjusted by the user. During the preprocessing stage, the quantile function (e.g., a linear function) is also added to the last layer of the DNN model.

C. Training DNN Model and Quantile Function

As depicted in Figure 3(b), the underlying DNN model and the attached quantile function are trained as a single network using the DNN training data. Then, during the back-propagation, the pinball loss is used to figure out the loss of the network. Specifically, the native DNN model makes a single-point estimate for each input, which is then passed to the quantile layer to determine the upper and lower boundaries of the PI. Then the loss of the network is computed as follows:

$$L_\alpha = \begin{cases} \alpha_{\text{tra}}(y - \hat{y}) & y - \hat{y} > 0 \\ (1 - \alpha_{\text{tra}})(\hat{y} - y) & \text{otherwise} \end{cases}, \tag{3}$$

where α_{tra} is a pre-defined mis-coverage rate (see Section IV-C) used at the training phase, y is the ground truth, and $\hat{y}$ is the single-point prediction given by the base DNN model.

Essentially, the pinball loss function applies different weights to positive and negative residuals based on a known confidence level $(1 - \alpha)$ [17]. A smaller value of α results in a greater penalty for samples with values smaller than the predicted value, and vice versa [45]. When α_{tra} is set to 0.5, Eq. 3 degenerates to the yielding median value of the corresponding dataset. Practically, setting α_{tra} to 0.1 expects that 90% of ground truths would fall within the generated PI.

D. Building Calibration Table

In Figure 3 (c), we outline constructing the calibration table using the left-over calibration dataset.

1) Calibration Table: The proposed calibration table is a matrix designed to record the residual errors of a sensor node (or location) for a given prediction window. Each row of the table stores the residuals of a specific sensor node for various prediction windows, while each column records the residuals of a particular prediction window for different sensor nodes.

2) Data Preprocessing: Similar to the standard DNN training procedure, where the training data is partitioned into training and validation sets, we split the calibration data into two sets: χ_1 for constructing the potential adjustments set for the initial PI and χ_2 for decision-making to determine the best adjustment. In the first set χ_1, the trained quantile DNN predicts the traffic information for each test node in a given prediction window. Then, the prediction errors or residuals are calculated by measuring how far the PI boundaries deviate from the ground truth. In the second validation set χ_2, the residual candidates obtained in χ_1 are used to determine the optimal values. These optimal values are stored in the corresponding cells of the calibration table for a given pair of node id and prediction window. We gain these optimal values by minimizing a carefully designed objective function. In this paper, we leave 50% of the calibration data as the χ_2.

3) Compute Residual Percentiles: To compute the residuals for populating the calibration table, we apply the trained quantile DNN to each test sample (from the first calibration dataset χ_1) of a given node i and prediction window to produce a PI, denoted as $\{\hat{y}^l_{i,j}, \hat{y}^u_{i,j}\}$, where $\hat{y}^l_{i,j}$ and $\hat{y}^u_{i,j}$ are the lower and the upper bound of the PI of node i in the prediction window j, respectively. We then compute the residuals of the PI as $R_{i,j} = \left|\hat{y}^u_{i,j} - y_{i,j}\right| \cap \left|\hat{y}^l_{i,j} - y_{i,j}\right|$, where $y_{i,j}$ is the true value of node i in the given prediction window j. Note that there may be multiple test samples for a given node-prediction window pair, resulting in n residuals denoted as $R^1_{i,j}, R^2_{i,j}, \ldots, R^n_{i,j}$.

Next, the percentiles of n residuals for each node-prediction-window combination could be constructed. Specifically, the residual candidates $R^1_{i,j}, R^2_{i,j}, \ldots, R^n_{i,j}$ are first sorted by value and subsequently partitioned into m equal-size groups, commonly referred to as percentiles, based on their relative position within the distribution. It is worth noting that the number of quantiles m is a user-defined parameter that balances calibration accuracy and computational complexity. For our study, we set $m = 100$. The 0th percentile corresponds to the lowest value of residuals, while the 100th percentile corresponds to the maximum value in the residual dataset. Compared to histogram-based approaches, residual quantiles demonstrate more robustness to outliers (as demonstrated in Section VII-F). Finally the residual percentiles are denoted as $Q^1_{i,j}, Q^2_{i,j}, \ldots, Q^m_{i,j}$.

4) Choosing the Best Quantile: After calculating the residual percentiles, the next step is to choose a specific quantile

from the distribution to be saved in the matching cell of a node-prediction-window pair. To this end, we apply the trained quantile DNN to test samples from the second validation set χ_2 to produce an initial PI for each test sample. Next, the residual quantiles, $Q^1_{i,j}, Q^2_{i,j}, \ldots, Q^m_{i,j}$, are applied to the initial PI to obtain an adjusted PI one by one. We determine which residual provides the best performance by comparing the coverage and effective width of each resulting PI for each node-prediction-window combination against the true value of the validation set. The optimal residual for the node-prediction-window combination is then recorded in the corresponding cell of the calibration table.

Specifically, our objective in selecting the residual recorded in the calibration table is to minimize the following:

$$L = \arg\min_{Q^m} \sum_{0}^{i} \sum_{0}^{j} -\lambda \mathcal{C}\text{ov}(Q^m_{i,j}, \chi_2) + (1-\lambda)\mathcal{PI}(Q^m_{i,j}, \chi_2), \quad (4)$$

where $\lambda \in [0, 1]$ is a weight to control the balance of the coverage $\mathcal{C}$ov and $\mathcal{PI}$ for a given node-prediction-window combination. Finally, the calibration table records the residual value where the index is the calibration quantile that gives the minimum L.

E. Using the Trained DNN and Calibration Component

Once we have trained the DNN model as well as the quantile function and constructed the calibration table as described above, we can then apply them to unseen test samples. This process is illustrated in Figure 2.

In contrast to standard CQR that applies a global adjustment to the initial PI, our approach utilizes the test input's sensor node id and prediction window to locate a residual value, δ_i, from a calibration table. Specifically, the trained DNN generates a single-point prediction for a given test sample that is then passed to the quantile function to create the initial PI, $\{l, u\}$. Using the node id of the test sample and the prediction window, we search the calibration table for the corresponding residual value δ_i. This value is used to amend the initial PI to the final PI as $\{l - \delta_i, u + \delta_i\}$, resulting in a wider PI for a positive δ_i and a narrower PI for a negative δ_i. No residual value may be presented in the calibration for a given combination of node id and prediction window, in which case the initial PI remains unchanged. Through this process, the PI can be more accurately adjusted to reflect the specific context of the prediction, leading to enhanced accuracy and better coverage.

Residual errors of missing combinations of node ids and prediction windows can be later inserted into the calibration table. This can be done by using the true value measured after the prediction window to compute the residual errors of the node-prediction-window pair. Similarly, during deployment or every time the DNN model is re-trained, the calibration table can be updated with the recorded predictions and true values. This way, the calibration table and quantile function can evolve to adapt to changes in the deployment environment.

TABLE I
DESCRIPTION OF THE DATASETS USED

Type	Dataset	No. sensors	Time period	Mean / Std
Speed	METR-LA	207	03/01/2012 - 06/30/2012	53.71 / 20.26
	PeMS-BAY	325	01/01/2017 - 05/31/2017	62.61 / 9.59
	PeMSD7(M)	228	05/01/2012 - 06/30/2012	58.89 / 13.48
Flow	PEMS03	358	09/01/2018 - 11/30/2018	179.3 / 143.7
	PEMS04	307	01/01/2018 - 02/28/2018	211.7 / 158.1
	PEMS07	883	05/01/2017 - 08/31/2017	308.5 / 188.2
	PEMS08	170	07/01/2016 - 08/31/2016	230.7 / 146.2

VI. EVALUATION SETUP

We evaluate QUANTRAFFIC on seven real-world datasets. In particular, we apply QUANTRAFFIC to representative DNN architectures for traffic forecasting and compare it with a wide range of baseline methods for uncertainty modeling. This section provides a detailed description of the datasets used in our evaluation, the baseline quantile methods that we compare QUANTRAFFIC against, and the hardware and software platforms used for the experiments.

A. Datasets

In our evaluation, we used seven public traffic datasets that are widely adopted in the literature. Table I provides the basics of each dataset, including the type of traffic data collected (e.g., speed or flow), the number of sensors used to collect the data, and the time covered by each dataset. The raw data from these datasets are aggregated into 5-minute intervals, in alignment with previous literature, e.g., [46].

B. Base Traffic Forecasting Models

To demonstrate the applicability of QUANTRAFFIC over a variety of DNN-based traffic predictors, we adopt five representative models, namely, Spatio-Temporal Graph Convolutional Network (STGCN) [18], Graph Wavenet (GWNet) [19], Graph Multi-Attention Network (GMAN) [20], Long-and Short-term Time-series Network (LSTNet) [21], and Multivariate Graph Neural Network (MTGNN) [22]. These models cover different variants of GNN and the long short-term memory (LSTM) architecture. They have been widely used in the literature and represent different classes of deep learning-based methods for traffic forecasting. Note that our goal is to demonstrate the generalization ability of QUANTRAFFIC to different models rather than to compare the relative performance of the base DNN architectures. Therefore, our evaluation is designed to compare the performance of various quantile methods in uncertainty modeling, all within the same base DNN architecture.

C. Competing Baselines

We compare QUANTRAFFIC against five representative uncertainty quantification methods and two classical methods based on historical data:

Historical data-based methods assume that traffic data, such as speeds or flow distribution, exhibit strong repeating patterns during the same period (e.g., the same day across

weeks or the same hour across days). These methods predict current travel speed and flow distribution by utilizing the distribution of the same period from prior data, such as the distribution of the same day from the previous weeks. In our experiments, we consider two baseline methods that rely on historical data: *Hist-D*, which uses data from the previous days in the training dataset for predictions, and *Hist-W*, which uses the distribution of the same period from the previous weeks for prediction. For example, if we were to predict the PI for node i at 8 a.m. on Monday, *Hist-W* would calculate the mean μ_i^t and variance σ_i^t across all previously seen samples of node i at 8 a.m. of all Mondays. Once the mean and variance for each time slot and node of interest are calculated, these two values can be used to construct the prediction interval for that slot and node with $\{\mu_i^t\text{-}\sigma_i^t,\ \mu_i^t + \sigma_i^t\}$.

Bayesian uncertainty quantification models the uncertainty in the model parameters using a likelihood function constructed by Bayesian modeling [24]. It also computes the data uncertainty by approximating the probability distribution over the model outputs through sampling and averaging over the resulting models. Our work uses a Gaussian prior distribution with zero mean and unit variance [24] and the MC sampling number of 50.

Monte Carlo dropout [25] models predictive distributions by randomly switching off neurons in DNNs during testing. This generates different model outputs that can be interpreted as samples from a probabilistic distribution, allowing MC dropout to estimate the prediction probability. Our work added a dropout layer with a rate of 0.3 after the last hidden layer of the base traffic forecasting model and used a sample of 50.

Deep quantile regression (DQR) generates PIs by using quantile function [26], [36]. Unlike conventional methods that minimize the averaged residual errors, DQR calculates the prediction errors at a specific distribution quantile. This method requires differential models or tailored loss objectives such as NLL-based or pinball losses to generate the boundaries of the PI. We use the 5th and 95th percentile estimates, following the configuration outlined in [16].

Conformal prediction [27] is a post-processing method for quantifying prediction uncertainties. The main idea behind conformal prediction is to use a nonconformity measure to evaluate the similarity between new input and training data. Given a certain confidence level, conformal prediction constructs prediction regions containing a certain fraction of predictions with the same nonconformity measure. In this study, we use inductive conformal prediction [47] due to its simplicity, which requires splitting the training data into two subsets. We use the same training dataset splitting ratio for conformal prediction and our approach.

D. Experimental Setting

1) Hardware and Software: We implemented QUANTRAFFIC as a Python package using PyTorch version 1.8.0. We conduct our model training and evaluation on a multi-core machine equipped with a dual-socket 20-core, 2.50 GHz Xeon(R) Silver 4210 CPU, 256GB of DDR4 RAM, and 2x NVIDIA GeForce RTX 2080 Ti GPUs. Our system is running Ubuntu 18.04.5 with Linux kernel 4.15.0, and we execute the GPU code using CUDA version 11.1.

2) Training Setup: For consistency with prior work [7], we split the data into training, validation, and test sets at a ratio of 7:1:2. To model uncertainty, we further reserve 40% of the training data for calibration (see Section V-B). We use the Kullback–Leibler divergence loss function to train Bayesian models [24], pinball loss for DQR, CQR, and the proposed QUANTRAFFIC. We use mean absolute error (MAE) as the loss function for all other models. We train all models using the Adam optimizer [48] with a learning rate of 0.001 and a batch size of 64 (except for GMAN, where we use a batch size of 16 due to GPU memory constraints). The training process stops after 200 training epochs or when the validation loss remains unchanged for ten consistent epochs, following the standard practice in prior work [7].

To ensure a fair comparison, the hyperparameters of quantile-based methods are identical to those of CQR methods. For the additional hyperparameters in CQR, i.e., calibration mis-coverage rate α_{cal} controlling the movement and direction of the upper and lower marginal bounds, we use the calibration set to pick the best value.

E. Evaluation Methodology

We vary the observation step from 1 to 12, corresponding to historical data in the last 5 to 60 minutes, as each step contains sensor data over 5 minutes. We also set the prediction windows to $1, 2, \ldots, 12$ steps by requiring the based DNN model to predict traffic information in the next $5, 10, \ldots, 60$ minutes. In other words, the base model takes as input sensor data of the last n minutes to predict information in the next n minutes.

We evaluate the coverage and discrimination guarantees when applying an uncertainty method to each tested DNN model. As explained in Section IV-C, we use MPIW (where smaller values are better) to evaluate the discriminative performance and PICP (where larger values are better) to quantify the coverage of the generated PIs for each test sample. We compute PICP and MPIW by averaging the results across nodes in the prediction window settings.

VII. EXPERIMENTAL RESULTS

We evaluate the effectiveness of QUANTRAFFIC on seven datasets, comparing it to six state-of-the-art uncertainty modeling methods, two classical methods based on historical data. Highlights of our evaluation are:

- QUANTRAFFIC consistently delivers the best overall performance than the baseline uncertainty methods across base DNN architectures and datasets (Section VII-A);
- The adaptive calibration scheme enhanced QUANTRAFFIC by giving a better trade-off between the coverage and PI width for individual locations and sensor nodes over CQR (Section VII-B and Section VII-F);
- QUANTRAFFIC gives more robust performance over CQR and DQR at different desired coverage rates (Section VII-C and Section VII-D);
- We showcase how QUANTRAFFIC can be used to enhance the evaluation of traffic forecasting models (Section VII-G).

TABLE II
PERFORMANCE EVALUATION FOR TRAFFIC METHODS WITH DIFFERENT UNCERTAINTY QUANTIFICATION METHODS ON TRAFFIC SPEED DATASETS. (PICP (%) ↑/MPIW ↓)

(a) METR-LA

Model Name	MC dropout	Bayesian	Conformal	DQR	CQR	QUANTRAFFIC
STGCN	−62.4% / 3.0	−44.9% / 7.1	−5.2% / 13.7	+0.5% / 30.1	+0.6% / 20.6	**+1.1% / 20.5**
GWNet	−47.9% / 3.5	−38.9% / 11.7	−4.5% / 11.9	−0.3% / 19.7	+1.2% / 15.6	**+1.8% / 15.1**
MTGNN	−61.6% / 2.4	−63.0% / 2.7	−4.3% / 11.8	−0.1% / 19.3	−0.4% / 14.7	**+0.1% / 14.6**
GMAN	−68.4% / 1.7	−11.9% / 16.1	−1.2% / 18.1	+0.9% / 27.4	+1.3% / 20.4	**+1.6% / 20.1**
LSTNet	−54.7% / 3.6	−66.0% / 3.0	−2.7% / 24.1	−0.1% / 29.0	+0.3% / 24.7	**+0.4% / 24.0**
AVERAGE	−59.00% / 2.84	−44.94% / 8.12	−3.58% / 15.92	+0.18 % / 25.10	+0.60% / 19.20	**+1.00% / 18.86**
Other baselines		Hist-D: +3.6% / 34.2		Hist-W: −9.2% / 28.2		DeepSTUQ: −0.4% / 19.3

(b) PEMS-BAY

Model Name	MC dropout	Bayesian	Conformal	DQR	CQR	QUANTRAFFIC
STGCN	−58.8% / 1.6	−38.1% / 3.7	−3.3% / 6.8	−2.4% / 9.8	−0.1% / 8.2	**+0.0% / 8.1**
GWNet	−47.3% / 2.9	−32.9% / 3.9	−1.8% / 5.9	−0.9% / 7.8	+0.1% / 6.3	**+0.6% / 6.2**
MTGNN	−66.4% / 1.2	−67.5% / 1.2	−1.6% / 5.8	−0.6% / 7.8	−0.2% / 6.2	**+0.2% / 6.1**
GMAN	−64.8% / 1.4	−3.3% / 7.3	−5.5% / 5.7	+0.5% / 9.7	−0.2% / 7.6	**+0.5% / 7.6**
LSTNet	−31.9% / 29.2	−84.0% / 0.3	−0.6% / 12.3	−5.1% / 10.5	−1.5% / 10.3	**−0.9% / 9.9**
AVERAGE	−53.84% / 7.26	−45.16% / 3.28	−2.56% / 7.30	−1.70% / 9.12	−0.38% / 7.72	**+0.08% / 7.58**
Other baselines		Hist-D: −13.8% / 9.5		Hist-W: −32.7% / 6.2		DeepSTUQ: +0.6% / 6.6

(c) PEMSD7M

Model Name	MC dropout	Bayesian	Conformal	DQR	CQR	QUANTRAFFIC
STGCN	−56.0% / 2.8	−36.7% / 5.8	−7.1% / 12.2	−2.4% / 17.0	+0.4% / 17.0	**+1.1% / 16.9**
GWNet	−40.1% / 3.6	−37.2% / 6.6	−4.6% / 9.6	−0.5% / 14.4	−0.2% / 13.0	**+0.1% / 12.9**
MTGNN	−37.8% / 4.6	−53.2% / 2.8	−2.2% / 11.6	−1.0% / 13.0	−0.2% / 13.0	**+0.2% / 12.9**
GMAN	−51.8% / 2.7	−2.1% / 14.4	−3.7% / 12.2	−0.7% / 17.0	+0.7% / 15.4	**+0.8% / 15.1**
LSTNet	−33.0% / 5.9	−16.2% / 9.6	−3.1% / 17.7	−4.8% / 16.7	−0.4% / 17.6	**+0.0% / 17.6**
AVERAGE	−43.74% / 3.92	−29.08% / 7.84	−4.14% / 12.66	+1.88% / 15.62	+0.06% / 15.20	**+0.44% / 15.08**
Other baselines		Hist-D: −15.3% / 10.7		Hist-W: −35.0% / 7.9		DeepSTUQ: +1.0% / 11.3

- We discuss how the data impact QUANTRAFFIC's performance in terms of training-calibration split ratio (Section VII-E) and pre-processing data splitting methods (Section VII-H).
- We present a real-world case study to demonstrate the practical application and effectiveness of QUANTRAFFIC (Section VII-I)

A. Overall Results

In this experiment, we set an expectation of a 90% coverage rate, and we evaluate if the quantification method can reach or even exceed the 90% coverage rate across test samples using PICP. Later in Section VII-D, we compare the performance under different coverage rate expectations.

Tables II and III present the results obtained when applying each evaluated uncertainty method to different DNN models and datasets. The best-performing results are highlighted in bold for each base DNN model. In the corresponding column of each uncertainty method, we first show the improvement of PICP (as a percentage concerning a 90% coverage rate), followed by the MPIW that measures the width of the PI (*lower-is-better*). In this context, a positive PICP improvement means that the coverage of PI exceeds the expectation of a 90% coverage rate. In contrast, a negative PCIP improvement (−) means that the coverage of PI falls below the expectation. For example, in Table II(a), applying QUANTRAFFIC to STGCN yields +1.1% / 20.5. This should read as that QUANTRAFFIC achieves a 91.1% coverage rate (above the coverage expectation) with 20.5 in MPIW (i.e., the difference between the upper and the lower bound of the PI is 20.5). We report the average PCIP and MPIW for each dataset across test samples. Other comparative baselines, including *Hist-W*, *Hist-D*, and DeepSTUQ, are shown at the bottom of each dataset table.

As expected, the MPIW given by all methods depends on the base DNN model's capability because a larger MPIW (or a wider PI) is needed to ensure the coverage for a less accurate DNN model. However, we observe that QUANTRAFFIC achieves or exceeds the desired 90% coverage rate for most test cases. For a handful of cases where the QUANTRAFFIC PICP falls below 90%, the resulting coverage rate remains close to the target of 90%, with a coverage rate of at least 88% (i.e., $\geqslant -1.8\%$ in the tables). In addition to achieving a good coverage rate, QUANTRAFFIC produces small MPIW, thus providing a meaningful uncertainty measure for decision-making.

Compared to QUANTRAFFIC, non-frequentist baselines such as MC dropout, Bayesian, and Conformal methods provide a narrow PI, resulting in a small MPIW. However, they struggle to meet the coverage expectation, where the true value

TABLE III
PERFORMANCE EVALUATION FOR TRAFFIC METHODS WITH DIFFERENT UNCERTAINTY QUANTIFICATION METHODS ON TRAFFIC FLOW DATASETS (PICP (%) ↑/MPIW ↓)

(a) PEMS03

Model Name	MC dropout	Bayesian	Conformal	DQR	CQR	QUANTRAFFIC
STGCN	−54.9% / 18.0	−18.1% / 57.9	−3.8% / 75.8	−10.8% / 61.8	−2.1% / 81.6	**−0.5% / 78.7**
GWNet	−36.0% / 30.4	−8.2% / 79.6	−1.5% / 64.3	−4.5% / 55.3	+1.5% / 66.2	**+1.2% / 66.1**
MTGNN	−27.3% / 40.9	−49.9% / 18.8	−0.8% / 65.7	−2.0% / 58.6	−0.5% / 60.6	**−0.4% / 61.6**
GMAN	−41.8% / 29.8	+5.7% / 127.1	−1.5% / 64.3	−8.5% / 67.8	−1.2% / 85.5	**−1.3% / 85.9**
LSTNet	−31.9% / 29.2	−57.8% / 40.4	−2.3% / 73.2	−4.0% / 63.3	+0.4% / 70.8	**+0.4% / 69.1**
AVERAGE	−38.38% / 29.66	−25.66% / 64.76	−1.98% / 68.66	−5.96% / 61.36	−0.38% / 72.94	**−0.12% / 72.28**
Other baselines		Hist-D: −27.4% / 67.6		Hist-W: −56.9% / 37.2		DeepSTUQ: −0.4% / 61.7

(b) PEMS04

Model Name	MC dropout	Bayesian	Conformal	DQR	CQR	QUANTRAFFIC
STGCN	−51.5% / 22.6	−19.4% / 64.1	+0.9% / 108.7	−4.1% / 86.7	−0.3% / 93.5	**+0.2% / 94.7**
GWNet	−39.0% / 37.5	−3.1% / 115.6	+0.5% / 92.5	−1.3% / 84.4	+0.9% / 85.5	**+1.1% / 85.3**
MTGNN	−28.1% / 55.0	−46.8% / 25.6	+0.5% / 91.3	−1.4% / 81.2	+0.5% / 81.2	**+0.8% / 82.4**
GMAN	−48.6% / 32.0	−0.2% / 116.4	+2.0% / 117.3	+5.6% / 132.7	+0.2% / 97.9	**+0.9% / 100.9**
LSTNet	−29.9% / 39.4	−52.6% / 25.4	+0.8% / 102.8	−0.3% / 92.8	+1.3% / 93.1	**+1.4% / 92.3**
AVERAGE	−39.42% / 37.30	−24.42% / 69.42	+0.94% / 102.52	−0.30% / 95.56	+0.52% / 90.24	**+0.88% / 91.12**
Other baselines		Hist-D: −21.8% / 92.2		Hist-W: −58.2% / 52.1		DeepSTUQ: +0.8% / 83.2

(c) PEMS07

Model Name	MC dropout	Bayesian	Conformal	DQR	CQR	QUANTRAFFIC
STGCN	−51.9% / 23.9	−18.4% / 88.4	−1.3% / 116.3	−8.4% / 94.9	−0.5% / 112.2	**+0.0% / 108.7**
GWNet	−80.6% / 5.9	−6.8% / 117.9	−1.3% / 93.5	−1.1% / 92.9	+0.4% / 94.9	**+0.4% / 92.5**
MTGNN	−52.6% / 24.8	−51.8% / 23.2	−2.6% / 88.3	−1.7% / 88.1	−0.3% / 90.7	**−0.1% / 89.1**
GMAN	−45.2% / 35.9	+0.7% / 145.6	−6.5% / 137.1	+4.5% / 154.7	−1.8% / 117.4	**−1.8% / 116.8**
LSTNet	−52.6% / 24.8	−53.1% / 30.3	−2.2% / 107.8	−1.4% / 107.9	−0.3% / 109.2	**+0.0% / 107.1**
AVERAGE	−56.58% / 23.06	−25.88% / 81.08	−2.78% / 108.60	−1.62% / 107.70	−0.50% / 104.88	**−0.30% / 102.84**
Other baselines		Hist-D: −20.7% / 110.2		Hist-W: −21.2% / 59.6		DeepSTUQ: +1.0% / 87.9

(d) PEMS08

Model Name	MC dropout	Bayesian	Conformal	DQR	CQR	QUANTRAFFIC
STGCN	−50.8% / 18.8	−15.8% / 65.4	−0.4% / 90.6	−5.9% / 74.4	−0.2% / 89.0	**+0.1% / 83.4**
GWNet	−37.1% / 36.3	−1.3% / 99.8	−0.7% / 70.6	+0.6% / 74.8	+0.1% / 75.1	**+0.1% / 72.0**
MTGNN	−26.9 % / 47.4	−35.3% / 27.8	−1.3% / 69.1	−1.2% / 68.5	+1.4% / 73.1	**+1.2% / 71.1**
GMAN	−39.3 % / 31.0	+0.9% / 104.7	−2.1% / 118.7	+1.2% / 92.6	+3.0% / 93.0	**+3.4% / 92.6**
LSTNet	−36.1 % / 28.2	+1.0% / 145.0	+0.5% / 85.1	−2.0% / 77.3	+0.7% / 80.3	**+0.8% / 78.3**
AVERAGE	−38.04% / 32.34	−10.10% / 88.54	−0.80% / 86.82	−1.46% / 77.52	+1.00% / 82.10	**+1.12% / 79.48**
Other baselines		Hist-D: −36.4% / 80.2		Hist-W: −29.8% / 41.9		DeepSTUQ: +1.0% / 71.2

of the test sample often falls outside the PI. This suggests that the PI given by non-frequentist baselines can mislead decision-making by being too optimistic or over-conservative for traffic forecasting. The performance of non-frequentist baselines is also not consistent, showing significant variance depending on the test datasets and DNN models. For instance, while Bayesian gives a coverage rate of 86.7% (−3.3% in Table II) on PEMS-BAY when using GMAN as the base DNN, its coverage rate is only 22.5% (−67.5% in Table II)) on the same dataset when using MTGNN as the base model.

By utilizing previously collected data, Hist-D gives good coverage but with a large MPIW, leading to over-conservative forecasts. In contrast, quantile-based methods such as DQR and CQR significantly outperform others by providing a better coverage guarantee. By incorporating an adaptive scheme to adjust the initial PI based on the test node and prediction window, QUANTRAFFIC improves upon CQR and DQR with higher PCIP and smaller MPIW for all test cases. QUANTRAFFIC also delivers consistent good performance across datasets and base DNN models, demonstrating the robustness of our approach. This further highlights the advantage of quantile-based methods over non-frequentist baselines, which exhibit inconsistent and varying performance depending on the test datasets and base DNN models.

QUANTRAFFIC also improves over DeepSTUQ, a state-of-the-art uncertainty modeling method, for the majority of the test cases. For example, for the PEMS04 dataset, under a PICP of 90.8%, QUANTRAFFIC gives a lower MPIW value than DeepSTUQ (82.4 verse 83.2). We emphasis that DeepSTUQ uses a particular DNN model architecture built upon Graph Convolutional Networks and Gated Recurrent Networks and are based on variational inference and deep ensembles,

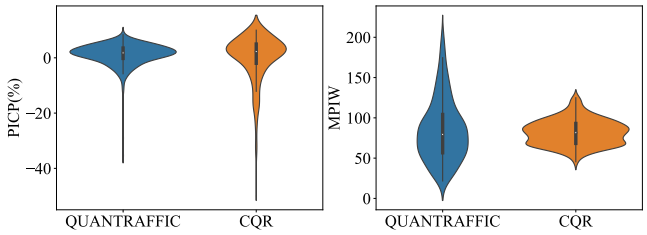


Fig. 4. Violin diagrams show PICP and MPIW by applying CQR and QUANTRAFFIC to GWNet model tested on the PEMS04. The thick black line shows where 50% of the data locates.

TABLE IV
PERFORMANCE COMPARISON OF QUANTILE METHODS UNDER DIFFERENT COVERAGE LEVELS FOR PICP (%) ↑/MPIW ↓

Expected Coverage	DQR	CQR	QUANTRAFFIC
0.6	−41.1% / 1.0	−0.1% / 4.1	**+0.1% / 4.1**
0.7	−33.0% / 2.0	−1.5% / 5.3	**−1.1% / 5.2**
0.8	−21.2% / 4.3	−1.7% / 7.8	**−1.4% / 7.6**
0.95	−5.2% / 21.5	−8.5% / 20.9	**−2.5% / 20.2**
Avg.	−30.09% / 5.76	−2.50% / 8.15	**−1.11% / 7.97**

hence incompatible with the underlying DNN models (e.g., STGCN, GWNet, etc.) evaluated in this work. In contrast, QUANTRAFFIC is a generic framework that can work with any underlying DNN models. Therefore, QUANTRAFFIC has a better generalization ability than DeepSTUQ.

B. Adaptive PI Adjustments

QUANTRAFFIC advances CQR by adaptively adjusting the initial PI through a dedicated calibration table that uses different adjustments rather than a constant global value adopted by standard CQR. To demonstrate the benefit of QUANTRAFFIC over CQR, we closely examine the test samples of the PEMS04 dataset when using GWNet as the base DNN for traffic forecasting. The results are given in the violin diagram of Figure 4. In the diagram, the width of the violin indicates the density of test data points with a given PICP, and the center of the plot is a box plot with the median and quartiles the test samples fall into a specific PICP (or coverage rate).

For this test scenario, the averaged PICP given by CQR and QUANTRAFFIC is comparable at 90.9% and 91.1%, respectively. However, the effectiveness of the PI can vary significantly for individual nodes, highlighting the importance of having an adaptive scheme. Upon closer inspection of Figure 4, we see that the coverage rates provided by QUANTRAFFIC are more uniformly located around the desired coverage rate of 90% across nodes. In contrast, the coverage rate provided by CQR is highly diverse, ranging from 45% to 100%. In other words, the performance of CQR is less consistent and less robust than QUANTRAFFIC, as CQR can lead to a poor coverage guarantee for individual nodes under a given prediction window.

C. Case Study of Selected Test Samples

In Figure 5, we closely examine the PI generated by different uncertainty methods for traffic flow prediction performed on PEMS04 on the date from 2012-02-18 18:00 to 2012-02-19 18:00. In the diagram, the PI is represented as a grey area, while the ground truth of a test sample is represented by a point. If a point falls within the grey belt, the generated PI covers the true value, otherwise, it fails to cover the true value. The grey area of a good strategy should cover as many points as possible while being as narrow as possible.

Methods like MC drop and Hist-W result in a small grey area with a small MPIW, but their PIs fail to cover the ground truth for a large number of test samples, leading to a poor PICP and insufficient coverage. Quantile methods like DQR and CQR perform significantly better than MC dropout and Bayesian. Compared to DQR and CQR, QUANTRAFFIC produces PIs that cover the ground truth of more test samples (i.e., a higher PICP) with a smaller MPIW (i.e., a narrower grey area in the diagram). This example demonstrates the effectiveness of our adaptive scheme.

D. Impact of Desired Coverage Rates

So far, our evaluation set 90% as the targeting coverage rate. In this section, we investigate the impact of the desired coverage rate on the performance of uncertainty methods. Previous evaluations have shown that DQR, CQR and QUANTRAFFIC are the best-performing methods. Therefore, we focus on quantile-based methods in this experiment. We apply the quantile methods to the METR-LA dataset using GWNet as the base DNN model. We vary the expected coverage level, $1-\alpha$, from 0.6 to 0.95, corresponding to a target coverage rate of 60% to 95%, respectively, to evaluate the usefulness of traffic uncertainty models in different practical scenarios.

Table IV compares the PCIP/MPIW given by each quantile method averaged across the test samples for a target coverage level. As expected, when we increase the coverage level to ensure a coverage rate, a larger MPIW (and a wider PI) is required. Once again, QUANTRAFFIC outperforms the other methods, providing the best coverage rate across settings. While DQR has the smallest MPIW in most cases, it gives a poor coverage rate, which can be up to 41% below the expected value. This suggests that the PI provided by DQR is too narrow (and hence has a small MPIW) to cover the true value of the prediction. CQR addresses the issues of DQR by using a constant adjustment value. However, it can still provide poor coverage or a PI that is too wide for some individual nodes. By dynamically adjusting the PI based on individual nodes, QUANTRAFFIC provides the best overall performance. This further demonstrates the effectiveness and flexibility of our approach in adapting to different sensor nodes or locations.

E. Impact of Training-Calibration Split Ratio

To build the calibration component, QUANTRAFFIC and CQR require setting aside some data from the training dataset as the calibration data. Our experiments described so far leave 40% of the original training data as the calibration dataset. In this experiment, we evaluate how the training-calibration dataset ratio affects the performance on QUANTRAFFIC and CQR. Specifically, We vary the ratio (β) between the DNN

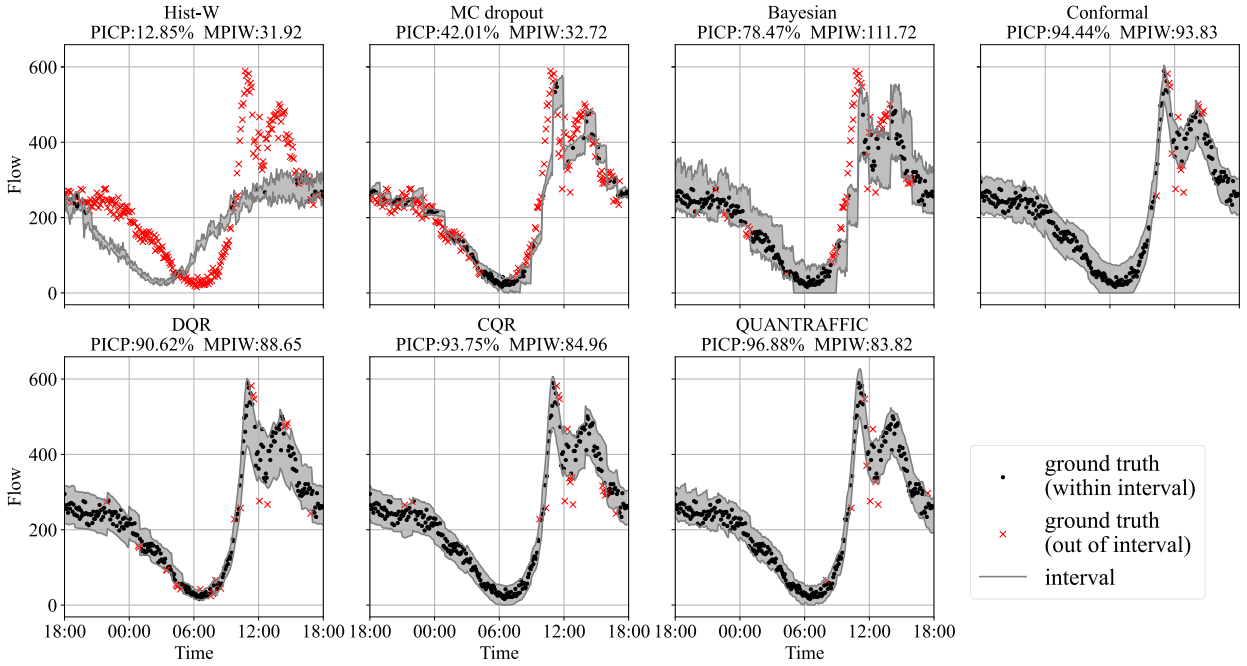


Fig. 5. Coverage and prediction intervals for different uncertainty quanlification models of PEMS04 from 2012-02-18 18:00 to 2012-02-19 18:00.

TABLE V

PERFORMANCE COMPARISON OF DIFFERENT TRAINING-CALIBRATION RATIO β ON TRAFFIC FLOW DATASET PEMS08 (PICP (%) ↑/MPIW ↓)

β	CQR	QUANTRAFFIC
1:6	+2.4% / **88.7**	**+3.0%** / 88.9
2:5	+0.5% / 76.2	**+0.8% / 76.1**
3:4	+0.1% / 72.5	**+0.1% / 72.3**
1:1	+0.3% / 74.6	**+0.5% / 73.6**
4:3	+2.6% / 73.9	**+2.6% / 71.7**
5:2	+1.4% / 73.6	**+1.6% / 73.4**
6:1	+2.1% / **74.3**	**+2.4%** / 75.1
Avg.	+1.35% / 76.25	**+1.56% / 75.88**

TABLE VI

PERFORMANCE COMPARISON OF GRID SEARCH AND QUANTILE SEARCH (PICP (%) ↑/MPIW ↓)

Frequency	Grid search	Quantile search
10	−49.1% / 31.5	**+0.9% / 73.1**
20	−23.8% / 44.7	**+0.7% / 72.0**
40	−9.9% / 56.3	**+0.5% / 71.4**
60	−5.6% / 61.9	**+0.4% / 71.1**
80	−3.7% / 64.5	**+0.4% / 70.9**
100	−2.7% / 66.3	**+0.4% / 70.9**

model training data and the calibration data to examine the impact on performance. Specifically, we evaluate the ratios of 6:1, 5:2, 4:3, 1:1, 3:4, 2:5, and 1:6. We then apply QUANTRAFFIC and CQR to the PEMS08 dataset, using GWNet as the base DNN model, with a target coverage rate of 90%.

The result is given in Section V. Leaving too few samples to train the base DNN model may lead to decreased accuracy, resulting in a wider PI (and larger MPIW) required to ensure sufficient coverage. For instance, when the training-calibration data ratio β is set to 1:6 or 2:5, CQR and QUANTRAFFIC produce a wide PI. However, insufficient calibration data can also impact the accuracy of the uncertainty model, given that QUANTRAFFIC relies on partitioning the calibration dataset to construct the calibration table. As a result, a smaller calibration dataset (e.g., when β is 6:1) can affect the performance of the model. Nevertheless, for most experimental settings, QUANTRAFFIC outperforms CQR in terms of both the PICP and MPIW metrics.

F. Quantile Search for Calibration Table Construction

As described in Section V-D, in order to construct the calibration table, we first apply the trained DNN and quantile function to the calibration dataset to obtain the prediction residuals. We then partition the residuals into continuous intervals with equal percentiles rather than predefined equal-sized intervals. This experiment compares our quantile-based approach against a grid method for projecting the residuals onto intervals. The results were obtained on the PEMS08 dataset using GWNet as the base DNN model and a target coverage rate of 90%. The results are given in Section VI, demonstrating the effectiveness of quantile search over the grid search method.

Using quantiles over a specific range can be advantageous in situations where traffic data is partially missing and the range of possible values for residuals is not well-defined. Quantile search reduces the need for dense interval steps by focusing the search on the most promising regions in the parameter space, thereby increasing the efficiency of the search process. It can also help to avoid overfitting and increase robustness to outliers. On the other hand, in grid search, missing traffic data can cause a narrow range of possible values, requiring more interval steps to capture better adjustments.

TABLE VII
PERFORMANCE EVALUATION BASED ON POINT AND UNCERTAINTY METRICS

Data	Method	Point estimation			Uncertainty estimation	
		MAPE ↓	RMSE ↓	MAE ↓	PICP ↑	MPIW ↓
METR-LA	STGCN	13.09%	11.81	5.08	90.13%	19.52
	GWNet	**11.93%**	**11.49**	**4.85**	90.14%	15.22
	MTGNN	12.17%	11.62	4.87	**90.24%**	**14.79**
	GMAN	14.10%	11.91	5.48	89.36%	17.27
	LSTNet	15.38%	11.89	6.34	89.83%	23.28
PEMS08	STGCN	11.15%	18.01	28.11	90.06%	83.44
	GWNet	**9.59%**	**15.05**	**23.81**	90.05%	72.01
	MTGNN	9.69%	15.15	23.98	**91.16%**	**71.13**
	GMAN	12.99%	18.29	28.06	90.20%	80.77
	LSTNet	10.86%	17.07	26.83	90.78%	78.31

G. Performance Evaluation of Traffic Forecasting Models

The ability to model prediction uncertainty can also be useful in evaluating the creditability of a traffic forecasting mode. In this experiment, we extend our evaluation to explore the trade-off between accuracy and confidence in traffic forecasting tasks. There is increasing research effort in discovering the best-scoring traffic predictor in specific or ideal scenarios [49]. Meanwhile, the reliability of the experimental evaluation is often neglected. For example, a traffic forecasting model could have a good performance on average in point estimation matrices but be less accurate during peak hours than at night. In this evaluation, we apply QUANTRAFFIC to METR-LA and PEMS08 and compare the prediction accuracy measured by commonly used loss functions, Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE) and the corresponding coverage and discrimination on test data measured in PICP and MPIW.

As can be seen from the Table VII, GWNet would be regarded as the most accurate model using traditional point-based evaluation metrics, but it is less reliable in coverage and discrimination metrics. While evaluating model credibility is not the focus of this work, our approach can provide a new measure for the performance evaluation of traffic forecasting models.

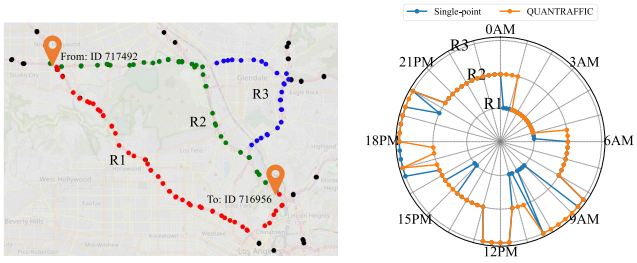
H. Impact of Pre-Processing Data Splitting Methods

So far, our evaluation is based on cross-validation on the timeline. In this section, we aim to explore the influence of varying timeline window lengths on the performance of uncertainty methods. To achieve this, we employed a rolling cross-validation technique comprising ten folds. The model was trained on consecutive folds and subsequently tested on the subsequent fold in an iterative manner. The final results were derived by averaging the outcomes obtained from all folds.

Table VIII presents the cross-validation performance of GWNet utilizing various uncertainty qualification methods on the PEMS08 traffic flow dataset. The analysis results align closely with those discussed in Section VII-A, where time series were split chronologically. Specifically, QUANTRAFFIC exhibits a high coverage rate of approximately 90% while maintaining a small MPIW. Moreover, QUANTRAFFIC demonstrates improved robustness, indicated by a smaller variance in both PICP and MPIW.

TABLE VIII
CROSS-VALIDATION PERFORMANCE EVALUATION FOR GWNET WITH DIFFERENT UNCERTAINTY QUANTIFICATION METHODS ON TRAFFIC FLOW DATASET PEMS08 (MEAN±VARIANCE)

	PICP(%) ↑	MPIW ↓
MC dropout	−33.1±3.8	38.7±5.2
Bayesian	+0.4±2.3	114.1±20.7
Conformal	−0.4±1.5	75.7±6.2
DQR	−1.9±1.5	70.8±6.6
CQR	+0.6±2.0	76.0±4.2
QUANTRAFFIC	**+1.2±1.1**	**76.0±2.2**



(a) Routes (b) Route selection

Fig. 6. Comparative analysis of route selection between single-point model and QUANTRAFFIC.

I. Case Study

We now consider a practical use case of QUANTRAFFIC, as defined in Figure 6. In this experiment, we use GWNet as the baseline single-point prediction model.

Figure 6(a) gives the sensor locations from the PEMS08 dataset, where each dot represents a sensor location. Figure 6(b) shows the differences in route selection given by a single-point prediction and QUANTRAFFIC on the journey from sensor IDs 717492 to ID 716956 on Monday, August 8, 2016. In Figure 6(b), the horizontal axis represents time slots, while the vertical axis represents the selected paths, R1, R2, and R3. The paths are arranged from the inner ring to the outer ring in decreasing order of distance: R1 (15.2 miles), R2 (14.1 miles), and R3 (13.9 miles). During peak hours, the actual congestion situation follows the order of R1 > R2 > R3. Each dot in Figure 6(b) represents a path selection strategy during different time slots, enabling a comparison between the single-point prediction method and the QUANTRAFFIC approach.

During peak hours (9 a.m. - 9:30 a.m., 6 p.m. - 8 p.m.), QUANTRAFFIC selects a longer but faster route (R3) as this route has less traffic and the fastest travel time. In contrast, a single-point prediction tends to select a shorter route (R1) despite the possibility of encountering congestion. In the early morning hours (1 a.m. - 5 a.m.), both methods prefer the shorter route (R1), and during other periods (5 a.m. - 9:30 a.m., 1 p.m. - 5 p.m., 9 p.m. - 12 a.m.), both methods show a preference for the middle-distance route (R2) to balance the excessive congestion and travel distance.

VIII. Discussion

Naturally, there is room for future work and improvement. We discuss a few points here.

A. Alternative Quantile Functions

In the quantile DNN model training (see Section V-C), we attach a linear layer to the last layer of the DNN and use the linear layer as the quantile function. An alternate solution could be using a dedicated network to generate two separate predictions. In this way, for each input, the DNN model would produce two separate outputs from the modified last layer, one for the lower bound and the other for the upper bound. The pinball loss assigns orthogonal weights to both predictions and uses them to calculate the loss. We leave this as our future work.

B. Calibration Component

In this paper, we use a dedicated calibration table to provide customized PI adjustments for individual nodes. Another way to do this is to train a calibration function using, e.g. linear regression or a neural network. Doing so would require having sufficient training data to learn an accurate calibration function.

C. Coverage Goal

QuanTraffic aims to produce a PI that meets the user-defined coverage level. In general, a higher coverage level guarantees stronger prediction accuracy. In practise, a high coverage level can be employed to ensure, for instance, the worst-case arrival time. In contrast, a lower coverage level may suffice if the user is willing to tolerate a certain level of prediction error in traffic information, for example, if the consequence of missing an event is insignificant. As such, techniques for learning and modeling user needs [50] are complementary to QuanTraffic.

D. Other Calibration Techniques

QuanTraffic employs a straightforward yet efficient method of utilizing a calibration table to adjust the PI. Additionally, other post-processing techniques can be used to enhance initial predictions. For instance, ensemble methods [37] can leverage multiple prediction models, and data augmentation techniques [9] involve applying various data augmentation methods to the test input to obtain multiple predictions to form the PI. Our future work will investigate the use of these techniques.

E. Data Distribution Drifts

Although differences in data characteristics between the training and calibration sets may lead to decreased accuracy in model prediction, the calibration component of QuanTraffic is used to adjust the prediction interval during calibration to address this drop in accuracy. However, in many real-world applications, particularly in traffic forecasting, the data distribution may be shifted due to various factors such as changes in road infrastructure (e.g., temporary road closures), weather patterns, and traffic patterns (e.g., traffic accidents), leading to a shift in the test data distribution with respect to the training data. In case the accuracy of QuanTraffic decreases, a user can re-train the base model to predict new intervals, update the uncertainty component, or integrate continuous learning methods [51] to periodically update the base model and the QuanTraffic component. Our future work will investigate how online adaptation techniques can be employed to detect data drift.

F. Calibration Data Size

QuanTraffic and CQR require setting aside some data from the model training dataset as calibration data. If the calibration set is not representative of the test set, the performance of CQR and QuanTraffic may suffer, as shown in Section VII-D. As such, techniques for data augmentation [52] like basic data augmentation methods (e.g. window cropping [53]), Deep Generative Models [54] and data selection [55] like active learning and scoring functions [56] are orthogonal to our approach.

IX. Conclusion

We have presented QuanTraffic, a framework that enhances the capability of DNN-based traffic forecasting models to quantify the uncertainty of their predictions. Specifically, QuanTraffic generates the upper and lower bounds of the prediction, which is useful in tasks like emergency route planning to ensure the worst-case arrival time. Our framework is generic, applicable to any DNN model, and does not alter the DNN model's underlying architecture during deployment. QuanTraffic builds on CQR and utilizes a dedicated loss function to train a quantile function that generates a prediction confidence interval for the single-point output of the DNN model. It advances standard CQR by dynamically adjusting the prediction interval based on individual locations or sensor nodes of the test samples, leading to a more accurate PI with valid coverage.

We evaluate QuanTraffic by applying it to six representative DNN architectures for traffic forecasting and comparing it against five uncertainty quantification methods and two classical historical-data-based methods. Experimental results show that QuanTraffic has good generalization ability, delivering better performance than the competing methods across the evaluated DNN models. This outcome underscores the significance of QuanTraffic as one of the first attempts to develop a generic framework for modeling uncertainties in traffic forecasting.

We hope that the open-source release of QuanTraffic will enable more research into robust traffic forecasting by providing a means to quantify uncertainty in DNN-based models.

References

[1] M. Akhtar and S. Moridpour, "A review of traffic congestion prediction using artificial intelligence," *J. Adv. Transp.*, vol. 2021, pp. 1–18, 2021.

[2] Y. Ye, Y. Zhu, C. Markos, and J. J. Q. Yu, "CatETA: A categorical approximate approach for estimating time of arrival," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 24389–24400, Dec. 2022.

[3] R. Tian, S. Li, and G. Yang, "Retraction note: Research on emergency vehicle routing planning based on short-term traffic flow prediction," *Wireless Pers. Commun.*, vol. 128, no. 2, pp. 1509–1509, Jan. 2023.

[4] Y. Yang and M. Ramezani, "A learning method for real-time repositioning in E-hailing services," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 1644–1654, Feb. 2023.

[5] D. A. Tedjopurnomo, Z. Bao, B. Zheng, F. M. Choudhury, and A. K. Qin, "A survey on modern deep neural network for traffic prediction: Trends, methods and challenges," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 4, pp. 1544–1561, Apr. 2022.

[6] X. Yin, G. Wu, J. Wei, Y. Shen, H. Qi, and B. Yin, "Deep learning on traffic prediction: Methods, analysis, and future directions," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 4927–4943, Jun. 2022.

[7] R. Jiang et al., "Dl-Traff: Survey and benchmark of deep learning models for urban traffic prediction," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, 2021, pp. 4515–4525.

[8] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *J. Big Data*, vol. 6, no. 1, pp. 1–54, Dec. 2019.

[9] S. Thulasidasan, G. Chennupati, J. A. Bilmes, T. Bhattacharya, and S. Michalak, "On mixup training improved calibration and predictive uncertainty for deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–12.

[10] H. Hong et al., "Heteta heterogeneous information network embedding for estimating time of arrival," in *Proc. 26th ACM SIGKDD Int. Knowl. Discovery Data Mining*, 2020, pp. 2444–2454.

[11] S. A. A. Kohl et al., "A probabilistic U-net for segmentation of ambiguous images," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 6965–6975.

[12] Z. Zhou, Y. Wang, X. Xie, L. Qiao, and Y. Li, "Stuanet understanding uncertainty in spatiotemporal collective human mobility," in *Proc. Web Conf.*, New York, NY, USA, 2021, pp. 1868–1879.

[13] Y. Li, L. Luo, H. Lin, H. Chen, and P.-A. Heng, "Dual consistency semi supervised learning with uncertainty quantification for COVID 19 lesion segmentation from CT images," in *Proc. Med. Image Comput. Comput. Assist. Intervent (Miccai)*, 2021, pp. 199–209.

[14] J. M. Hoque, G. D. Erhardt, D. Schmitt, M. Chen, and M. Wachs, "Estimating the uncertainty of traffic forecasts from their historical accuracy," *Transp. Res. A, Policy Pract.*, vol. 147, pp. 339–349, May 2021.

[15] W. Qian, D. Zhang, Y. Zhao, K. Zheng, and J. J. Q. Yu, "Uncertainty quantification for traffic forecasting: A unified approach," in *Proc. 39th Int. Data Eng.*, 2023, pp. 1–6.

[16] Y. Romano, E. Patterson, and E. J. Candès, "Conformalized quantile regression," in *Proc. 33rd Int. Neural Inf. Process Syst.*, 2019, pp. 3543–3553.

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

[18] B. Yu, Y. Haoteng, and Z. Zhanxing, "Spatio temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. 27th Int. Joint Artif. Intell.*, 2018, pp. 3634–3640.

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

[20] C. Zheng, X. Fan, C. Wang, and J. Qi, "GMAN: A graph multi-attention network for traffic prediction," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 1, pp. 1234–1241, Apr. 2020.

[21] G. Lai, W. Chang, Y. Yang, and H. Liu, "Modeling long and short term temporal patterns with deep neural networks," in *Proc. Sigir*, 2018, pp. 95–104.

[22] 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. Knowl. Discovery Data Mining*, 2020, pp. 753–763.

[23] V. Vovk, A. Gammerman, and G. Shafer, *Algorithmic Learning in a Random World*. New York, NY, USA: Springer, 2005.

[24] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," in *Proc. 32nd Int. Conf. Int. Mach. Learn.*, 2015, pp. 1613–1622.

[25] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation representing model uncertainty in deep learning," in *Proc. 33rd Int. Conf. Mach. Learn.*, vol. 48, M. F. Balcan and K. Q. Weinberger, Eds. New York, NY, USA, Jun. 2016, pp. 1050–1059.

[26] R. Koenker and G. Bassett Jr., "Regression quantiles," *Econometrica, J. Econ. Soc.*, vol. 46, no. 1, pp. 33–50, 1978.

[27] G. Rhafer and V. Vovk, "A tutorial on conformal prediction," *J. Mach. Learn. Res.*, vol. 9, pp. 371–421, Mar. 2008.

[28] G. E. P. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *J. Amer. Stat. Assoc.*, vol. 65, no. 332, pp. 1509–1526, Dec. 1970.

[29] N. Ramakrishnan and T. Soni, "Network traffic prediction using recurrent neural networks," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2018, pp. 187–193.

[30] W. Jiang and L. Zhang, "Geospatial data to images: A deep-learning framework for traffic forecasting," *Tsinghua Sci. Technol.*, vol. 24, no. 1, pp. 52–64, Feb. 2019.

[31] S. C. Chau, S. Shen, and Y. Zhou, "Decentralized ride-sharing and vehicle-pooling based on fair cost-sharing mechanisms," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 1936–1946, Mar. 2022.

[32] Z. Chen, S. Wu, S. Shen, Y. Liu, F. Guo, and Y. Zhang, "Co-optimization of velocity planning and energy management for autonomous plug-in hybrid electric vehicles in urban driving scenarios," *Energy*, vol. 263, Jan. 2023, Art. no. 126060.

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

[34] T. Pearce, A. Brintrup, M. Zaki, and A. Neely, "High quality prediction intervals for deep learning a distribution free ensembled approach," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, J. Dy and A. Krause, Eds. Jul. 2018, pp. 4075–4084.

[35] D. J. C. MacKay, "A practical Bayesian framework for backpropagation networks," *Neural Comput.*, vol. 4, no. 3, pp. 448–472, May 1992.

[36] Y. Chung, W. Neiswanger, I. Char, and J. Schneider, "Beyond pinball loss quantile methods for calibrated uncertainty quantification," in *Proc. 35th Int. Conf. Neural Inf. Process Syst.*, 2021, pp. 10971–10984.

[37] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proc. Annu. Neural Inf. Process Syst.*, 2017, pp. 6402–6413.

[38] H. Ritter, A. Botev, and D. Barber, "A scalable Laplace approximation for neural networks," in *Proc. 6th Int. Conflearn Represent.*, 2018, pp. 1–15.

[39] W. Huang, J. Zhang, and K. Huang, "Bootstrap estimated uncertainty of the environment model for model based reinforcement learning," in *Proc. 33rd AAAI Artif. Intell.*, 2019, pp. 3870–3877.

[40] K. Stankeviciute, A. M. Alaa, and M. van der Schaar, "Conformal time series forecasting," in *Proc. Adv. Neural Inf. Process Syst.*, vol. 34, 2021, pp. 6216–6228.

[41] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intell. Syst.*, vol. 13, no. 4, pp. 18–28, Jul./Aug. 1998.

[42] H. Papadopoulos and H. Haralambous, "Reliable prediction intervals with regression neural networks," *Neural Netw.*, vol. 24, no. 8, pp. 842–851, Oct. 2011.

[43] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[44] Z. Lin, S. Trivedi, and J. Sun, "Locally valid and discriminative prediction intervals for deep learning models," in *Proc. 35th Int. Conf. Neural. Inf. Process Syst.*, 2021, pp. 1–14.

[45] Q. Wang, Y. Ma, K. Zhao, and Y. Tian, "A comprehensive survey of loss functions in machine learning," *Ann. Data Sci.*, vol. 9, no. 2, pp. 187–212, Apr. 2022.

[46] S. Guo, Y. Lin, H. Wan, X. Li, and G. Cong, "Learning dynamics and heterogeneity of spatial–temporal graph data for traffic forecasting," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 11, pp. 5415–5428, Nov. 2022.

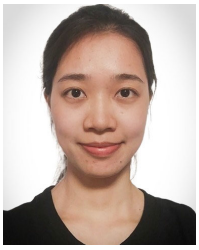
[47] H. Papadopoulos, "Inductive conformal prediction: Theory and application to neural networks," in *Tools in Artificial Intelligence*, P. Fritzsche, Ed. Rijeka, Croatia: IntechOpen, 2008, ch. 18, doi: 10.5772/6078.

[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, Y. Bengio and Y. LeCun, Eds. May 2015, pp. 1–15.

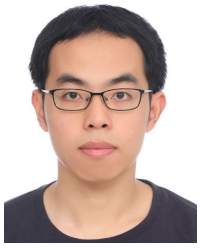
[49] I. Laña, J. J. Sanchez-Medina, E. I. Vlahogianni, and J. D. Ser, "From data to actions in intelligent transportation systems: A prescription of functional requirements for model actionability," *Sensors*, vol. 21, no. 4, p. 1121, Feb. 2021.

[50] G. I. Webb, M. J. Pazzani, and D. Billsus, "Machine learning for user modeling," *User Model. User-Adapted Interact.*, vol. 11, pp. 19–29, Mar. 2001.

[51] M. De Lange et al., "A continual learning survey: Defying forgetting in classification tasks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3366–3385, Jul. 2022.
[52] Q. Wen et al., "Time series data augmentation for deep learning a survey," in *Proc. 30th Int. Joint Artif. Intell.*, Aug. 2021, pp. 4653–4660.
[53] A. Le Guennec, S. Malinowski, and R. Tavenard, "Data augmentation for time series classification using convolutional neural networks," in *Proc. Ecml Pkdd Workshop Adv. Analytics Learn. Temporal Data*, 2016, pp. 1–9.
[54] T. E. K. Lee, Y. Kuah, K.-H. Leo, S. Sanei, E. Chew, and L. Zhao, "Surrogate rehabilitative time series data for image based deep learning," in *Proc. 27th Eur. Signal Process Eusipco*, 2019, pp. 1–5.
[55] S. Eetemadi, W. Lewis, K. Toutanova, and H. Radha, "Survey of data-selection methods in statistical machine translation," *Mach. Transl.*, vol. 29, nos. 3–4, pp. 189–223, Dec. 2015.
[56] M. Eck, S. Vogel, and A. Waibel, "Low cost portability for statistical machine translation based on N-gram frequency and TF-IDF," in *Proc. 2nd Int. Workshop Spoken Lang. Transl.*, 2005, pp. 1–7.



Ying Wu is currently pursuing the joint Ph.D. degree with the Southern University of Science and Technology and the University of Leeds. Her research interests include data mining, intelligent transportation systems, and deep learning.



Yongchao Ye (Student Member, IEEE) received the B.Eng. degree in computer science and technology from Ningbo University, Ningbo, China, in 2020, and the M.S. degree in electrical science and technology from the Southern University of Science and Technology, Shenzhen, China, in 2023. His research interests include spatio-temporal data mining in smart cities, intelligent transportation systems, and federated learning.

Adnan Zeb is currently a Post-Doctoral Fellow with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. His research interests include deep representation learning, deep graph embedding networks, spatio-temporal data mining, and their applications to intelligent transportation systems.

James Jianqiao Yu (Senior Member, IEEE) is currently a Lecturer with the Department of Computer Science, University of York, and an Honorary Assistant Professor with the Department of Electrical and Electronic Engineering, The University of Hong Kong. His work is also mainly on forecasting, decision-making, and privacy preservation of future transportation systems and artificial intelligence techniques for industrial applications. His research interests include smart cities and privacy computing, deep learning, intelligent transportation systems, and smart energy systems. He is an Editor of *IET Smart Cities*. He was the World's Top 2% Scientists from 2020 to 2023 and of career by Stanford University, ranked at the top 0.3% of all artificial intelligence scholars.

Zheng Wang is currently a Professor of intelligent software technology with the School of Computing, University of Leeds. He works to make software development easier and more accessible so that every programmer can easily write, maintain, and optimize software.