cST-ML: Continuous Spatial-Temporal Meta-Learning for Traffic Dynamics Prediction

Yingxue Zhang*, Yanhua Li*, Xun Zhou†, Jun Luo‡
*Worcester Polytechnic Institute
†University of Iowa
‡Lenovo Group Limited
yzhang31@wpi.edu; yli15@wpi.edu; xun-zhou@uiowa.edu; jluo1@lenovo.com

***Abstract*—Urban traffic status (*e.g.*, traffic speed and volume) is highly dynamic in nature, namely, varying across space and evolving over time. Thus, predicting such traffic dynamics is of great importance to urban development and transportation management. However, it is very challenging to solve this problem due to spatial-temporal dependencies and traffic uncertainties. In this paper, we solve the traffic dynamics prediction problem from Bayesian meta-learning perspective and propose a novel continuous spatial-temporal meta-learner (cST-ML), which is trained on a distribution of traffic prediction tasks segmented by historical traffic data with the goal of learning a strategy that can be quickly adapted to related but unseen traffic prediction tasks. cST-ML tackles the traffic dynamics prediction challenges by advancing the Bayesian black-box meta-learning framework through the following new points: 1) cST-ML captures the dynamics of traffic prediction tasks using variational inference; 2) cST-ML has novel designs in architecture, where CNN and LSTM are embedded to capture the spatial-temporal dependencies between traffic status and traffic related features; 3) novel training and testing algorithms for cST-ML are designed. We also conduct experiments on two real-world traffic datasets (taxi inflow and traffic speed) to evaluate our proposed cST-ML. The experimental results verify that cST-ML can significantly improve the urban traffic prediction performance and outperform all baseline models.**

***Index Terms*—traffic dynamics prediction, Bayesian meta-learning, spatial-temporal data.**

I. Introduction

Over past a few decades, the rapid population growth has accelerated the process of urbanization, which in turn brings huge impacts on the urban traffic including the increasing traffic volume, worse traffic condition and the overload of the transportation infrastructures. As a result, accurately predicting the *highly dynamic traffic status* (*e.g.*, traffic volume, speed and inflow) has become a crucial work for urban development aiming to reduce congestion and increase mobility, since it can not only provide insights for urban planning, help to improve the efficiency of public transportation, but also guarantee the public safety [20].

Given the underlying road network and the historical traffic observations, *the problem of traffic dynamics prediction* aims at forecasting short-term traffic status in consecutive time slots. However, there are many practical challenges before solving this problem:

1) *Spatial-temporal dependencies.* It is the most common challenge when dealing with traffic dynamics prediction problem, since the traffic status would be influenced by the nearby environments, road networks and its previous traffic status.

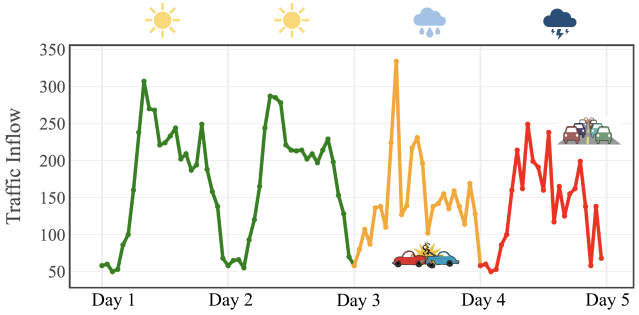


Fig. 1: Illustration of traffic dynamics.

2) *Traffic dynamics and temporal uncertainties.* In traffic dynamics prediction, the most difficult part is to capture and model the dynamics of traffic status, since urban traffic always contains temporal uncertainties due to sudden travel demand changes, unexpected events or extreme weather. For example, Figure 1 is an illustration of traffic dynamics, where it is possible that the traffic patterns are almost consistent in the first two days but show obvious fluctuations and temporal uncertainties in the next few days. The reasons of such considerable changes in traffic patterns could be a thunder storm, a large sports event or a car crash. In general, irregular and drastic traffic changes caused by these factors are hard to capture using traditional time series models due to their non-periodicity and rareness (*i.e.*, lacking training samples).

A lot of research efforts have been put into the traffic dynamics prediction area. Some works use traditional machine learning methods and time series models to predict urban traffic. Works such as [2], [14] and [3] use support vector regression (SVR) to capture the relationships between traffic and environmental features. Another work [9] presents a traffic prediction method which combines the SARIMA model and multi-input autoregressive (AR) model with genetic algorithm (GA) optimization. In addition, deep neural networks are also widely used in urban traffic prediction works. For example, works [10] and [18] predict travel demands and traffic accidents using autoencoders and ConvLSTM, respectively. Other works including [17] and [4] combine CNN and LSTM to predict the traffic speed and crowd flows. However, these models do not consider the situation where the traffic shows

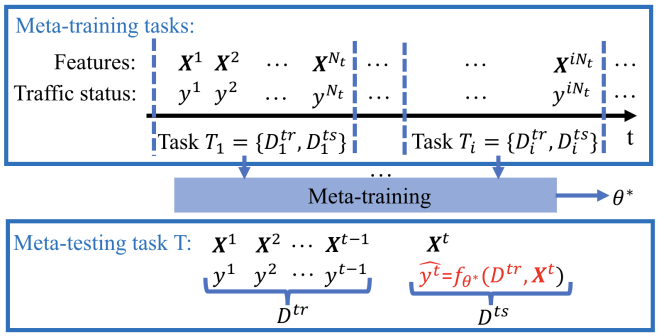


Fig. 2: Problem illustration.

strong non-stationarity.

In this paper, we try to solve the short-term traffic dynamics prediction problem and tackle the unique challenges mentioned before from the Bayesian meta-learning perspective. A novel continuous spatial-temporal meta-learner (cST-ML) is proposed, which is trained on a distribution of traffic prediction tasks generated by traffic time series data with the goal of learning a strategy that can be quickly generalized to related but unseen traffic prediction tasks from the same task distribution. cST-ML captures the spatial-temporal dependencies of traffic as well as the temporal uncertainties and dynamics through variational inference. Our **main contributions** are summarized as follows:

- We are the first to solve the traffic dynamics prediction problem from the Bayesian meta-learning perspective and propose a novel continuous spatial-temporal meta-learner cST-ML. cST-ML advances the Bayesian black-box meta-learning framework to capture traffic dynamics and temporal uncertainties. (See Sec III-A.)
- cST-ML features some novel designs in the architecture. cST-ML is composed of an inference network and a decoder, where CNN and LSTM are embedded to realize the goal of capturing traffic spatial-temporal dependencies. Novel algorithms are also designed for cST-ML training and testing. During meta-training and testing. (See Sec III-B and Sec III-C.)
- We conduct extensive experiments on two real-world traffic datasets (taxi inflow and traffic speed) to evaluate our proposed cST-ML. The experimental results verify that cST-ML can significantly improve the urban traffic prediction performance and outperform all existing baseline methods on both datasets. (See Sec IV.)

II. Overview

In this section, we define the traffic dynamics prediction problem, and outline our solution framework.

A. Problem definition

In a city, *urban traffic status* can be characterized by many statistics, such as traffic volumes, speed, inflow/outflow, *etc*, which are of great interests to urban planners and researchers for transportation planning, traffic evaluation and more. These statistics are dynamic in nature, namely, varying across space and evolving over time. Hence, we divide an urban area into grid cells as defined below. Each grid cell represents a target area for *urban dynamics prediction*.

Definition 1 (Grid cells). We divide a city into $I \times J$ grid cells with equal side-length (*e.g.*, $1 \times 1km$), denoted as $\mathcal{S} = \{s_{ij}\}$, where $1 \leq i \leq I, 1 \leq j \leq J$.

Definition 2 (Target region for a target grid cell). For a target grid cell s_{ij}, its target region is a square geographic region with s_{ij} in center, formed by $\ell \times \ell$ grid cells, denoted with $R_{ij} = \langle s_{ij}, \ell \rangle$.

In our study, we assume the traffic status in a target grid cell s_{ij} has high spatial correlations with the other grid cells within its target region.

Definition 3 (Traffic related features). All features that will influence the traffic status are traffic related features, *e.g.*, time of the day, travel demand, *etc*. For a grid cell s, we denote x^t as one feature of s in time slot t. For a target region R, we denote X^t as one feature map of R in time slot t, where X^t is a $\ell \times \ell$ matrix. Since there could be multiple features, all the feature maps in region R in time slot t can be denoted with a tensor $\boldsymbol{X}^t = \{X_1^t, \ldots, X_n^t\} \in \mathbb{R}^{n \times \ell \times \ell}$, where $n \in \mathbb{N}^+$ is the number of features.

Definition 4 (Traffic status). Traffic status indicates the quality of traffic, which can be measured by traffic inflow/outflow, average driving speed, *etc*. We denote y^t as the average traffic status of grid cell s in time slot t.

In this paper, we choose one specific measure of traffic status as the target of prediction, other measures if available can be treated as traffic related features during prediction.

Definition 5 (Traffic prediction task.) A traffic prediction task $\mathcal{T}_i$ is composed of a set of paired $(\boldsymbol{X}^t, y^t)$ in N_t consecutive time slots, which is divided into a training set $\mathcal{D}_i^{\text{tr}}$ and a testing set $\mathcal{D}_i^{\text{ts}}$, *i.e.*, $\mathcal{T}_i = \{(\boldsymbol{X}_i^1, y_i^1), \ldots, (\boldsymbol{X}_i^{N_t}, y_i^{N_t})\} = \{\mathcal{D}_i^{tr}, \mathcal{D}_i^{ts}\}$.

Problem Definition. For a specific target grid cell s, given all the historical traffic data, we aim to predict the traffic status $\{\hat{y}^t\}$ in consecutive time slots based on the available traffic related features $\{\boldsymbol{X}^t\}$. Since our goal is using meta-learning to solve this problem, the problem is transformed as follows: in meta-learning setup, the historical time series traffic data is segmented into τ tasks, we assume all the tasks are sampled from the same distribution, $\mathcal{T}_i \sim p(\mathcal{T})$. During meta-training, we aim to train a meta-learner (with parameters θ) whose objective is to minimize the expected loss with respect to θ over all training tasks sampled from $p(\mathcal{T})$:

$$\theta^\star = \arg\min_\theta \mathbb{E}_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}\left(\phi_i, \mathcal{D}_i^{\text{ts}}\right), \text{and } \phi_i = f_\theta\left(\mathcal{D}_i^{\text{tr}}\right). \quad (1)$$

During meta-testing, the meta-learner is evaluated on unseen testing tasks from the same task distribution. When predicting the future traffic, which can be view as a new testing task, we have:

$$\hat{y}^t = f_{\theta^\star}\left(\mathcal{D}^{\text{tr}}, \boldsymbol{X}^t\right), \quad (2)$$

where $\mathcal{D}^{\text{tr}} = \{(\boldsymbol{X}^1, y^1), \ldots, (\boldsymbol{X}^{t-1}, y^{t-1})\}$ includes a few training data in the current task. The problem is illustrated in Figure 2.

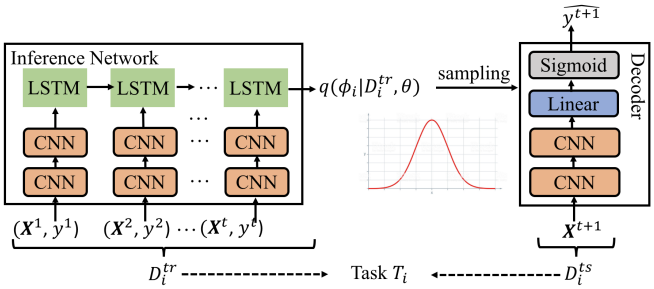


Fig. 3: cST-ML structure.

III. Methodologies

In this section, we introduce our continuous spatial-temporal meta-learning framework.

A. cST-ML Modeling

To capture the spatial-temporal dependencies and temporal uncertainties of tasks, now we are in a position to develop continuous spatial-temporal meta-learning (cST-ML) framework, which advances the Baysian black-box meta-learning framework and design unique structures for meta-learner.

Following the original Bayesian black-box meta-learning [5], to deal with the uncertainties in task adaptation, we treat the adapted parameters as a latent variable. We approximate the likelihood with variational lower bound (ELBO), the ELBO is derived in Eq. 3.

$$\begin{aligned}\log p(x) &\geq \mathbb{E}_{q(z|x)}[\log p(x,z)] + H(q(z|x)) \\ &= \mathbb{E}_{q(z|x)}[\log p(x|z)] - D_{KL}(q(z|x)\|p(z)),\end{aligned} \tag{3}$$

where z is the latent variable and x is the real data, D_{KL} is the Kullback-Leibler divergence, $p(x|z)$ can be treated as an decoder and $q(z|x)$ is the inference network, $p(z) \sim N(0,1)$.

In traffic dynamics prediction, to advance the Bayesian black-box meta-learning framework and take uncertainties within tasks into consideration, we first segment the historical traffic data into τ tasks, each task is directly divided into $\mathcal{D}_i^{\text{tr}}$ and $\mathcal{D}_i^{\text{ts}}$ and applying cST-ML only once.

In this situation, for task $\mathcal{T}_i$, we have specific $\mathcal{D}_i^{\text{tr}}$ and $\mathcal{D}_i^{\text{ts}}$. Eq.4 is the log likelihood lower bound of the task $\mathcal{T}_i$:

$$\begin{aligned}L_\theta(\mathcal{T}_i) = L_\theta(\phi_i, \mathcal{D}_i^{\text{ts}}) = \mathbb{E}_{q(\phi_i|\mathcal{D}_i^{\text{tr}},\theta)}\left[\log p\left(y_i^{\text{ts}}|\boldsymbol{X}_i^{\text{ts}}, \phi\right)\right] \\ -D_{KL}\left(q\left(\phi_i|\mathcal{D}_i^{\text{tr}},\theta\right)\|p(\phi_i|\theta)\right),\end{aligned} \tag{4}$$

where q is the inference network and parameterizes the mean and log-variance diagonal of a Gaussian distribution, and ϕ_i is sampled from this distribution for each rolling window, the Kullback-Leibler divergence can be approximated using the reparameterization trick (see more information in [8]). Compared with Eq.3, the latent variable corresponds to the adapted parameter ϕ_i, and the information we use to infer ϕ_i includes $\mathcal{D}_i^{\text{tr}}$ and θ.

Thus, in traffic dynamics prediction problem, the final objective is to maximize the log likelihood lower bound across all meta-training tasks:

$$\max_\theta \mathbb{E}_{\mathcal{T}_i}[L_\theta(\mathcal{T}_i)]. \tag{5}$$

Algorithm 1 Meta-Training

Input: Task distribution $p(\mathcal{T})$, initialized cST-ML f_{θ_0}.
Output: Well trained cST-ML.

```
while not done do
    Sample a batch of tasks from p(T).
    for for each task T_i in the batch do
        Prepare D_i^tr and D_i^ts for each task in T_i.
        Sample φ_i from q(φ_i|D_i^tr, θ).
        Compute log likelihood using Eq.4.
    end for
    Update θ with Adam [7] to maximize Eq.5.
end while
```

Algorithm 2 Meta-Testing

Input: A new task $\mathcal{T} = \{(\boldsymbol{X}^1, y^1), \ldots, (\boldsymbol{X}^{t-1}, y^{t-1})\}$ with available $\boldsymbol{X}^t, \ldots, \boldsymbol{X}^{N_t}$, well-trained cST-ML $f_{\theta^\star}$.
Output: Predicted values $\{\hat{y}^t, \ldots, \hat{y}^{N_t}\}$.

```
Define D^tr = {(X^1, y^1), ..., (X^{t-1}, y^{t-1})} and D^ts = {X^t, ..., X^{N_t}}
for each X^t in D^ts do
    ŷ^t = f_{θ*}(D^tr, X^t).
end for
```

B. cST-ML Architecture

We also design unique structures for cST-ML to tackle the complex spatial-temporal traffic dependencies. The structure of our cST-ML is composed of an inference network and a decoder. The inference network tries to encode the training data within a task into a latent distribution which captures the spatial patterns of the current location and also learns the temporal dependencies and uncertainties, the decoder is responsible for the prediction using the testing data within the same task. Figure 3 shows the overall structure of cST-ML.

The Inference Network is CNN and LSTM based and is actually the adaptation process of a task, which takes in the $\mathcal{D}_i^{\text{tr}}$ and extracts information from $\mathcal{D}_i^{\text{tr}}$, aiming to output a latent distribution which captures uncertainties of $\mathcal{T}_i$. The input of the inference network is $\mathcal{D}_i^{\text{tr}} = \{(\boldsymbol{X}_i^1, y_i^1), \ldots, (\boldsymbol{X}_i^t, y_i^t)\}$, where $t < N_t$.

Since $\boldsymbol{X}^t$ is a tensor for each time slot, y^t is first enlarged to an $\ell \times \ell$ matrix and concatenates with $\boldsymbol{X}^t$, and then the concatenated tensor goes through a few layers of CNN activated by ReLU which can capture the spatial dependencies of local traffic. The output sequence is then fed into the LSTM, the hidden state of LSTM in the last time slot t goes through fully-connected layers and produces the mean and log variance of a Gaussian distribution $q\left(\phi_i|\mathcal{D}_i^{\text{tr}},\theta\right)$.

The Decoder aims to produce the prediction $\hat{y}^{ts}$ based on $\boldsymbol{X}^{ts}$ where $(\boldsymbol{X}^{ts}, y^{ts}) \in \mathcal{D}_i^{\text{ts}}$, the prediction loss is calculated using y^{ts} and $\hat{y}^{ts}$. Decoder takes two inputs, i) the adapted information ϕ_i sampled from $q\left(\phi_i|\mathcal{D}_i^{\text{tr}},\theta\right)$ and ii) $\boldsymbol{X}^{ts}$. $\boldsymbol{X}^{ts}$ first goes through a few layers of CNN activated by ReLU and then concatenates with ϕ_i, the results pass fully-connected layers activated by Sigmoid function and we get the final

prediction $\hat{y}^{ts}$. The detailed structure of the inference network and decoder are illustrated in Figure 3.

C. cST-ML Training and Testing

The detailed meta-training process is shown in Algorithm 1. We repeatedly sample tasks from the task distribution, for one batch tasks, we compute the total log likelihood and update θ once.

After training, the well-trained meta-learner θ can fast adapt to any new tasks. The meta-testing algorithm is shown in Algorithm 2. In meta-testing, to predict the future traffic, we define a new testing task $\mathcal{T} = \{(\boldsymbol{X}^1, y^1), \ldots, (\boldsymbol{X}^{t-1}, y^{t-1})\}$, we use $\mathcal{T}$ as training data to get future predictions of $\hat{y}^t$.

IV. Evaluation

In this section, we conduct extensive experiments on real-world traffic datasets to evaluate our cST-ML. We first describe the datasets and introduce experiments, then we present baselines compared with our model and the evaluation metrics. Finally, the experiment results are presented and analyzed in detail.

A. Dataset Descriptions

Preprocessing of Dataset

We evaluate our model on the real-world datasets including (1) traffic speed, (2) taxi inflow and (3) travel demand, all of which are extracted from Shenzhen, China from Jul 1st to Dec 31st. In the preprocessing step, we first apply map gridding to the whole Shenzhen City, where the city is partitioned into 40×50 grid cells, for each target grid cell, its target region is the 5×5 matrix with the target grid cell in center. Thus, there are in total $1,656$ possible target grid cells.

Traffic speed, taxi inflow and travel demand are all extracted from taxi GPS records collected in Shenzhen, China from Jul 1st to Dec 31st, 2016. In each time slot (*i.e.*, one hour) of each day, taxi inflow is the number of taxis that stay or arrive at a target grid cell, travel demand is the number of taxi pickups within a target grid cell. In effect, it is hard to obtain the travel demands of all transport modes in a target grid cell, thus, we use taxi demands to represent travel demands, and many studies have shown that taxi demand is a very representative measure of travel demand [6], [13].

Experiment Descriptions

Next, we describe our two traffic prediction experiments we will perform in detail.

- **Traffic speed prediction.** In speed prediction, the traffic status in each grid cell is measured by average traffic speed, and there are 12 time slots per task, *i.e.*, $N_t = 12$, and thus 184 tasks over 6 months. All the tasks are divided into meta-training tasks (the first 80% of all tasks) and meta-testing tasks (the rest of 20%). We treat travel demands, traffic inflow and the time of the day as traffic related features, and use meta-training tasks to train the model and use meta-testing tasks to do evaluations. The goal of this task is to predict the traffic speed of a target grid cell s based on the historical available features.
- **Taxi inflow prediction.** Similar to traffic speed prediction, in the taxi inflow prediction, the traffic status in each grid cell is measured by taxi inflow. We view travel demands, traffic speed and the time of the day as traffic related features. There are also 12 time slots per task, *i.e.*, $N_t = 12$, and 184 tasks over 6 months. All the tasks are divided into meta-training tasks (the first 80% of all tasks) and meta-testing tasks (the rest of 20%). We aim to train the model with meta-training tasks and evaluate the model using meta-testing tasks.

B. Baselines

- **HA [16]** For each grid cell, Historical Average method (HA) predicts the traffic status for a target grid cell based on its average status of the previous time slots.
- **Regression [1].** This method applies ridge regression to predict the future traffic status, the predictors are the corresponding traffic related features. The training data are used to train the regression model and the testing data are used for evaluations.
- **ARIMA [15].** Auto-Regressive Integrated Moving Average (ARIMA) is a conventional parametric based time-series model. Here we view the historical traffic status as time-series data and apply ARIMA to predict the future traffic status.
- **LSTM [12], [19].** This method uses LSTM to predict the future traffic status using traffic related features as input. The daily traffic related features can be viewed as an input sequence, which goes through CNNs first and then passes LSTM to get the predicted traffic status sequence.
- **SNAIL [11].** It is an state-of-the-art deterministic black-box meta-learning method. SNAIL utilizes attention layers to get the deterministic adapted parameters for each task instead of sampling from a distribution, where the task uncertainties are not considered.

C. Evaluation Metrics

We use mean absolute percentage error (MAPE) and rooted mean square error (RMSE) for evaluations:

$$\text{MAPE} = \frac{1}{T}\sum_{t=1}^{T} \left| y_t - \hat{y}_t \right| / y_t, \tag{6}$$

$$\text{RMSE} = \sqrt{\frac{1}{T}\sum_{t=1}^{T} \left(y_t - \hat{y}_t \right)^2}, \tag{7}$$

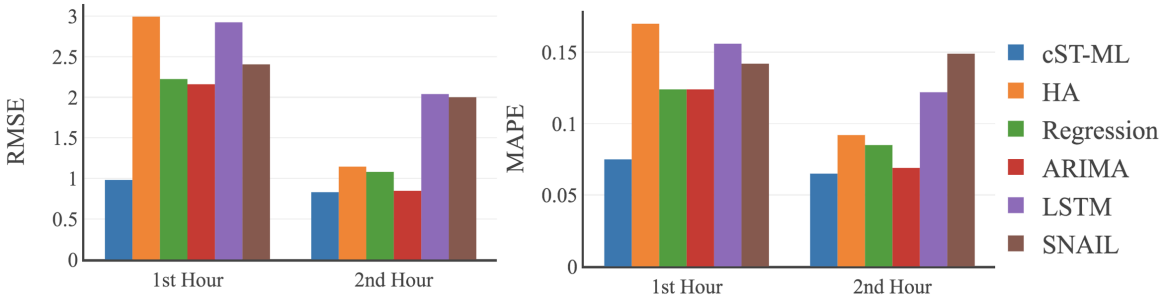
where y_t is the ground-truth traffic status observed in the target grid cell s in the t-th time slot, and $\hat{y}_t$ is the corresponding prediction, T is the total number of time slots to perform prediction.

D. Experimental Settings

The whole Shenzhen city is divided into 40×50 grid cells with a side-length $l_1 = 0.0084°$ in latitude and $l_2 = 0.0126°$ in longitude. The target region for a target grid cell is of size 5×5, *i.e.*, $\ell = 5$. Thus, there are in total $1,656$ possible target

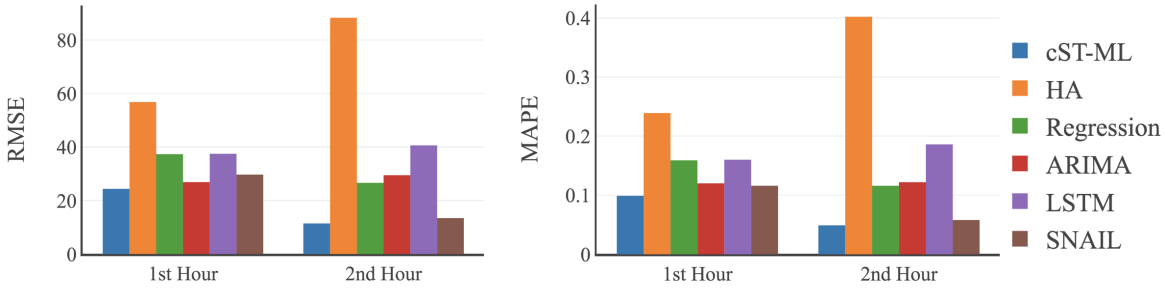
TABLE I: Performance on traffic speed prediction and taxi inflow prediction.

Methods			HA	Regression	ARIMA	LSTM	SNAIL	**cST-ML**
Traffic speed	1h	RMSE	2.993	2.224	2.160	2.923	2.405	**0.981**
		MAPE	0.170	0.124	0.124	0.156	0.142	**0.075**
	2h	RMSE	1.389	1.115	2.188	1.853	2.278	**0.915**
		MAPE	0.107	0.082	0.147	0.100	0.114	**0.070**
Taxi inflow	1h	RMSE	56.833	37.356	26.902	37.501	29.715	**24.376**
		MAPE	0.239	0.159	0.120	0.160	0.116	**0.099**
	2h	RMSE	79.37	35.496	27.259	31.992	26.179	**19.130**
		MAPE	0.327	0.119	0.108	0.121	0.102	**0.074**



(a) RMSE in 2 consecutive hours (b) MAPE in 2 consecutive hours

Fig. 4: Comparisons of models in 2 consecutive hours in traffic speed prediction.



(a) RMSE in 2 consecutive hours (b) MAPE in 2 consecutive hours

Fig. 5: Comparisons of models in 2 consecutive hours in taxi inflow prediction.

grid cells in Shenzhen city. In the experiment, we can select any possible target grid cell to perform traffic predictions.

The time interval for each task used to train the cST-ML are from 7:00am to 7:00pm, where each hour is a time slot and we have 12 time slots per day/task, *i.e.*, $N_t = 12$. Thus, we have $\mathcal{T}_i = \{(\boldsymbol{X}_i^1, y_i^1), \ldots, (\boldsymbol{X}_i^{12}, y_i^{12})\}$. In experiment, $\mathcal{D}_i^{\text{tr}} = \{(\boldsymbol{X}_i^1, y_i^1), \ldots, (\boldsymbol{X}_i^1, y_i^{10})\}$, $\mathcal{D}_i^{\text{ts}} = \{(\boldsymbol{X}_i^{11}, y_i^{11}), (\boldsymbol{X}_i^{12}, y_i^{12})\}$

The structure of cST-ML is as follows: two layers of CNN are utilized before LSTM in the inference network, the input channel of the first CNN is 4, the output channel is 64, the kernel size is 3, stride is 1 and padding is 1; for the second CNN layer, the input channel is 64, the output channel is 128, the kernel size is 5, stride is 1 and padding is 0. In decoder, we still use two layers of CNN combined with a linear transformation. cST-ML is trained using Adam optimizer [7] with $\beta_1 = 0.5$ and $\beta_2 = 0.999$, and a learning rate of 2×10^{-4} for 2000 times task samplings.

E. Evaluation Results

1) Average prediction performance: First, we conduct experiments to compare the average prediction performance of our proposed cST-ML and five competing baseline models. The results are shown in Table I. In the table, for a specific target grid cell, we present the RMSE and MAPE results for one-hour and two-hour traffic speed prediction and taxi inflow prediction. For meta-based models (including cST-ML and SNAIL), we randomly pick 5 meta-testing tasks in both of the traffic speed and taxi inflow predictions, compute the one-hour and two-hour RMSE and MAPE for each testing task and report the average results in the table. For other models, we use the same testing data to compute the statistics.

In traffic speed prediction, according to the average RMSE and MAPE in one-hour and two-hour predictions, cST-ML outperforms all the baseline models.

SNAIL is a deterministic black-box meta-learning method which does not consider any uncertainties of tasks, but it achieves competitive performance in some cases (*i.e.*, two-hour traffic speed and taxi inflow predictions), the reason is that we view daily traffic as one task in meta-training and meta-testing, for one specific target grid cell, in most of cases, the everyday traffic is similar which means there is less task uncertainties, and thus SNAIL can achieve competing prediction performance sometimes.

LSTM is used as a seq2seq model in traffic prediction, which utilizes the traffic related features to predict the traffic status, so it does not rely on the previous traffic status in testing

or prediction process which could result in larger prediction errors.

Compared with the traditional traffic prediction baseline models including HA, Regression and ARIMA, cST-ML achieves significant improvements since it not only captures the traffic spatial-temporal dependencies but also the temporal uncertainties. On the contrary, these traditional models only consider either the temporal dependencies or the relationships between traffic status and features, and they cannot deal with the traffic uncertainties very well.

In taxi inflow prediction, we get similar prediction results. SNAIL is the most competitive baselines compared with other baseline models, which indicates they can better learns the spatial-temporal patterns of traffic, and thus obtain lower errors. However, cST-ML is more powerful due to its novel designs.

2) Detailed performance in consecutive time slots: In this part, we are aiming to prove the effectiveness of our cST-ML in traffic predictions in each time slot (*e.g.*, one hour). In urban traffic prediction problem, the good average prediction performance is not enough, since we expect to get more accurate prediction for each specific time slot. Thus, we conduct experiments and provide detailed prediction performance for each time slot (*i.e.*, one hour). The statistics are calculated based on 5 meta-testing tasks in both of the traffic speed and taxi inflow predictions, in each time slot, we report the average RMSE and MAPE of all 5 testing tasks.

In traffic speed prediction, the detailed performance is presented in Figure 4. As shown in Figure 4(a) and Figure 4(b), our cST-ML achieves the best prediction performance, the performance of baselines including ARIMA and SNAIL presents higher volatilities and thus the prediction performance is much more unstable.

In taxi inflow prediction, as shown in Figure 5(a) and Figure 5(b), our cST-ML also achieves the best prediction performance, similar to Figure 4, the performance of all baseline models still presents much higher volatilities in prediction performance, in contrast, cST-ML displays more stable and accurate predictions in general, which also proves that cST-ML can better capture the traffic uncertainties and complex spatial-temporal dependencies, therefore, cST-ML provides more accurate and stable traffic prediction in consecutive time slots.

V. Conclusion

In this paper, we solved the traffic dynamics prediction problem using Bayesian meta-learning framework. We proposed a novel continuous spatial-temporal meta-learner (cST-ML), which learned a general traffic dynamics prediction strategy from historical traffic data (segmented into tasks) and could be quickly adapted to new prediction tasks containing just a few samples and exhibited excellent prediction performance. cST-ML captured the traffic spatial-temporal dependencies and the traffic uncertainties through new features in both objective and architecture beyond the original Bayesian black-box meta-learning. We conduct experiments on real-world traffic datasets (taxi inflow and traffic speed) to evaluate our proposed cST-ML. The experiment results verify that cST-ML can significantly improve the urban traffic prediction performance and outperforms all baseline models.

Acknowledgment

Yingxue Zhang and Yanhua Li were supported in part by NSF grants IIS-1942680 (CAREER), CNS-1952085, CMMI-1831140, and DGE-2021871. Xun Zhou is funded partially by Safety Research using Simulation University Transportation Center (SAFER-SIM). SAFER-SIM is funded by a grant from the U.S. Department of Transportation's University Transportation Centers Program (69A3551747131). However, the U.S. Government assumes no liability for the contents or use thereof.

References

[1] I. Alam, D. M. Farid, and R. J. F. Rossetti. The prediction of traffic flow with regression analysis. In *IEMIS*, pages 661–671, 2019.

[2] M. Castro-Neto, Y.-S. Jeong, M.-K. Jeong, and L. Han. Online-svr for short-term traffic flow prediction under typical and atypical traffic conditions. *Expert Systems with Applications*, 36:6164–6173, 2009.

[3] Y. Cong, J. Wang, and X. Li. Traffic flow forecasting by a least squares support vector machine with a fruit fly optimization algorithm. *Procedia Engineering*, 137:59 – 68, 2016.

[4] Z. Cui, R. Ke, and Y. Wang. Deep stacked bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction. In *6th International Workshop on Urban Computing*, 2017.

[5] C. Finn. Bayesian meta-learning. https://cs330.stanford.edu/slides/cs330_bayesian_metalearning.pdf, 2019. [Online].

[6] E. J. Gonzales, C. J. Yang, E. F. Morgul, and K. Ozbay. Modeling taxi demand with gps data from taxis and transit. Technical report, Mineta National Transit Research Consortium, 2014.

[7] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[8] D. P. Kingma and M. Welling. Auto-encoding variational bayes, 2013.

[9] X. Luo, L. Niu, and S. Zhang. An algorithm for traffic flow prediction based on improved sarima and ga. *KSCE Journal of Civil Engineering*, 22:1–9, 05 2018.

[10] Y. Lv, Y. Duan, W. Kang, Z. Li, F.-Y. Wang, et al. Traffic flow prediction with big data: A deep learning approach. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):865 – 873, 2015.

[11] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel. A simple neural attentive meta-learner. In *International Conference on Learning Representations*, 2018.

[12] O. Mogren. C-RNN-GAN: continuous recurrent neural networks with adversarial training. *CoRR*, 2016.

[13] N. Mukai and N. Yoden. Taxi demand forecasting based on taxi probe data by neural network. In *IIMSS*, pages 589–597, 2012.

[14] Y. Sun, B. Leng, and G. Wei. A novel wavelet-svm short-time passenger flow prediction in beijing subway system. *Neurocomputing*, 166, 04 2015.

[15] M. Tan, S. C. Wong, J. Xu, Z. Guan, and P. Zhang. An aggregation approach to short-term traffic flow prediction. *IEEE Transactions on Intelligent Transportation Systems*, pages 60–69, 2009.

[16] H. Yao, Y. Liu, Y. Wei, X. Tang, and Z. Li. Learning from multiple cities: A meta-learning approach for spatial-temporal prediction. In *The World Wide Web Conference*, page 2181–2191, 2019.

[17] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma. Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks. *Sensors*, 17(7), 2017.

[18] Z. Yuan, X. Zhou, and T. Yang. Hetero-ConvLSTM: A deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data. In *KDD*, pages 984–992, 2018.

[19] J. Zhao, F. Deng, Y. Cai, and J. Chen. Long short-term memory - fully connected (lstm-fc) neural network for pm2.5 concentration prediction. *Chemosphere*, 2019.

[20] Y. Zheng, L. Capra, O. Wolfson, and H. Yang. Urban computing: concepts, methodologies, and applications. *TIST*, 2014.