DAC-ML: Domain Adaptable Continuous Meta-Learning for Urban Dynamics Prediction

Xin Zhang
Worcester Polytechnic Institute
xzhang17@wpi.edu

Yanhua Li
Worcester Polytechnic Institute
yli15@wpi.edu

Xun Zhou
University of Iowa
xun-zhou@uiowa.edu

Oren Mangoubi
Worcester Polytechnic Institute
omangoubi@gmail.com

Ziming Zhang
Worcester Polytechnic Institute
zzhang15@wpi.edu

Vincent Filardi
Worcester Polytechnic Institute
vfilardi@wpi.edu

Jun Luo
Lenovo Group Limited
jluo1@lenovo.com

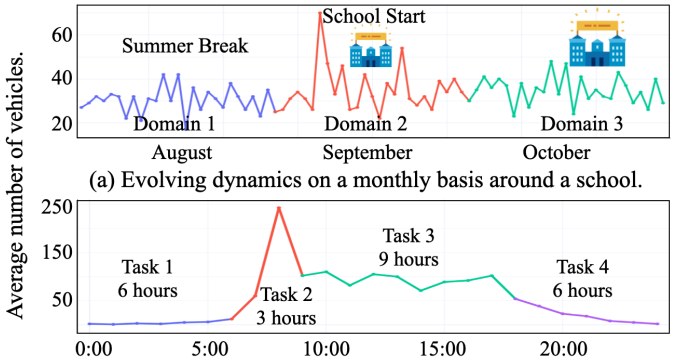
***Abstract*—Given the underlying road network of an urban area, the problem of urban dynamics prediction aims to capture the patterns of urban dynamics and to forecast short-term urban traffic status continuously from the historical observations. This problem is of fundamental importance to urban traffic management, planning, and various business services. However, predicting urban dynamics is challenging due to the highly dynamic (*i.e.*, varying across geographical locations and evolving over time) and uncertain (*i.e.*, affected by unexpected factors) nature of urban traffic systems. Recent works adopt meta-learning approaches to capture irregular and rare patterns but make unrealistic assumptions such as single-domain uncertainties and explicit temporal task segmentation. In this paper, we solve the urban dynamics prediction problem from the Bayesian meta-learning perspective and propose a novel *domain adaptable continuous meta-learning* approach (DAC-ML) that does not require task segmentation. Trained on a sequence of spatial-temporal urban dynamics data, DAC-ML aims to detect and infer unobserved latent variations (from task and domain levels) and generalize well in a sequential prediction setting, where the underlying data generating process varies over time. Experimental results on three real-world datasets demonstrate that DAC-ML can outperform baselines in urban dynamics prediction, especially when obvious urban dynamics and temporal uncertainties are present.**

***Index Terms*—Spatial-temporal data mining, meta-learning, domain adaptation**

I. Introduction

In an urban traffic system, "urban dynamics" refers to the overall mobility of humans and their interactions with the system. Given the underlying road network of an urban area, *the problem of urban dynamics prediction* aims to capture the patterns of urban dynamics and to forecast short-term urban traffic status (*e.g.*, traffic speed, volume) continuously from the historical observations obtained from IoT devices (*e.g.*, GPS equipment on vehicles and automated fare collection devices on buses and trains).

The urban dynamics prediction problem is of fundamental importance to many aspects of modern urban administration, such as urban transit planning, resource allocation, traffic management, and public safety [1], [2]. In addition, precise urban dynamics predictions (*e.g.*, customer flow and traffic condition) are also the basis of many location-based services, including but not limited to routing, package delivery, and ride-sharing.



(a) Evolving dynamics on a monthly basis around a school.

(b) Underlying task undergoes discrete and unobserved switches.

Fig. 1: Traffic dynamics with varied temporal granularity.

Though extensively studied in the literature as the traffic prediction problem, urban dynamics prediction is indeed more complex and challenging for modern cities. Due to the unprecedented travel demand surges accompanying the fast global urbanization process, the existing transportation infrastructures are increasingly overloaded, causing more frequent congestion and higher risks of traffic anomalies (*e.g.*, accidents). Also, unexpected human activities (*e.g.*, gathering, protests) have become an integrated part of modern urban dynamics. All such factors significantly increase the uncertainties of urban traffic, leading to previously unknown irregular patterns and making precise prediction a more difficult task.

State-of-the-art. Numerous research works have studied urban dynamics prediction, including methods based on traditional machine learning [3]–[6] and approaches applying deep neural networks [7]–[12]. However, most of these works assume that urban traffic dynamics follow regular or previously seen patterns in the historical data, and ignore uncertainties due to abrupt and uncontrollable factors such as extreme weather, accidents, and unexpected events. A few recent works try to capture such uncertainties from the perspective of meta-learning [13]–[15], due to its ability to adapt to new, unseen patterns quickly. Pan *et al.* [13] attempt to use a meta-graph

attention network to extract spatial correlation, and a meta-recurrent neural network for temporal dependency on traffic prediction. Yao *et al*. [14] view different cities as different training tasks, and propose to use the meta-learned knowledge from multiple cities to predict traffic status. A recent work, cST-ML [15], covers urban uncertainties with Bayesian meta-learning [16]. However, two practical limitations prevent these works from accurately predicting urban dynamics: *(i)* these meta-learning based approaches assume that urban dynamics evolve under the same pattern (*i.e*., domain) over time, and *(ii)* their test time adaption is achieved by explicitly segmenting continuous data into predefined discrete time windows (*i.e*., prediction "tasks"). We demonstrate these two limitations with examples below.

Limitation 1 (Domain shift): *Urban dynamics uncertainties are complex and may occur from multiple domains in a high-dimensional space.* For example, Fig. 1(a) demonstrates the vehicle inflow dynamics (*i.e*., the average number of vehicles coming to a region) over three months, around a school. Fig. 1(a) shows that the traffic pattern changes significantly over months, *i.e*., with high inflows in September when school starts and a low traffic inflow in August during the summer break. Such an example illustrates that the urban dynamics pattern shifts monthly. Similarly, the pattern shifts may also occur on a seasonal and weekly basis, affected by human activities and other factors. These drastic urban dynamic pattern changes indicate that they are from different "domains". Therefore, it is necessary to capture the domain information which embeds the evolving urban dynamics and temporal uncertainties, and view them from a high-dimensional space, *e.g*., various temporal granularities (daily, weekly, monthly, and seasonally) as different dimensions of a domain.

Limitation 2 (Data pattern with variable task lengths): *Urban dynamics data comes continuously as a sequence, and the underlying pattern changes unexpectedly without notice [1], [17].* These characteristics make the task separation assumption in previous works [13]–[15], [18] incompatible with the continuous prediction setting. Therefore, in the meta-learning formulation of the urban dynamics prediction problem, a "task" is to predict the urban dynamics in the next time step using all the data in the recent time interval that follows the same pattern. Essentially, the switches between these tasks are usually unobserved, and the length of each task is non-deterministic. For example, in Fig. 1(b), different tasks are colored differently, and the tasks do not have a consistent and deterministic length. Hence, pre-segmenting the urban dynamics data into fixed-length tasks may break the natural patterns of the underlying traffic dynamics and lower the prediction accuracy.

Our approach. To address the aforementioned two unrealistic limitations made by the state-of-the-art works, in this paper, we make the first attempt to solve the unique urban dynamics prediction problem by considering multiple domains in a continuous setting on the basis of Bayesian meta-learning. A novel domain adaptable Bayesian meta-learning approach (DAC-ML) is proposed, which can adapt data, extract domain information continuously, and predict sequentially from unsegmented urban dynamics data. Our main contributions are summarized as follows:

- We are the first to consider and model the ever-evolving urban dynamics as diverse domains. With this, we propose to use a domain inference network to constantly monitor domain changes and extract domain information to enable domain adaptation. (See Section III-B.)
- We develop a novel domain adaptable Bayesian meta-learning (DAC-ML) framework under the continuous setting to fulfill the needs for online adaptation and prediction. DAC-ML advances the Bayesian black-box meta-learning framework to enable adaptation in an online learning setting. (See Section III-C.)
- We identify the practical challenges for DAC-ML, namely, the spatial-temporal correlation and complex temporal uncertainty from various granularities. For the first practical challenge, we utilize ConvLSTM [9] to capture the spatial-temporal information from historical data. For the second, we view different granularities as different dimensions of a domain, and train several independent domain dimension inference modules to capture temporal uncertainty in multiple granularities. (See Section III-D.)
- We perform extensive experiments on real-world spatial-temporal datasets (traffic speed, taxi inflow, and travel demand) to evaluate our proposed DAC-ML. The experimental results demonstrate that DAC-ML outperforms baseline methods, and the extracted domain statistics contribute to urban dynamics prediction accuracy. (See Section IV.) **We make our code and unique dataset available to contribute to the research community[1].**

II. Overview

In this section, we define the urban dynamics prediction problem, and highlight the research challenges.

A. Preliminaries

Urban dynamics encompasses many aspects, including traffic speed, vehicle inflow/outflow, human mobility, *etc*. These aspects and their statistics, which vary across different geographical locations and evolve over time, can represent and characterize the urban dynamic status of a city. We divide an urban area into grid cells defined below to get a clearer view of the urban dynamics in different geographical locations. Suppose each grid cell is a target, we are able to study its urban dynamics influenced by its spatial and temporal neighbors.

Definition 1 (A grid cell s_{ij}). We partition a city into $I \times J$ grid cells, where each grid cell has equal side-length (*e.g*., $1 \times 1km^2$), denoted as $\mathcal{S} = \{s_{ij}\}$, where $1 \leq i \leq I$, $1 \leq j \leq J$.

Definition 2 (A target region R_{ij}). Each target grid cell s_{ij} comes with a target region R_{ij}. The target region R_{ij} is a square geographic region formed by $\ell \times \ell$ grid cells centering s_{ij}. We denote a target region as $R_{ij} = \langle s_{ij}, \ell \rangle$. In our study,

[1]The code and dataset are accessible at https://github.com/XinZhang525/DAC-ML.

we hold the assumption that the urban dynamics of each grid cell within a target region R_{ij} has strong spatial correlations, thus affecting the urban dynamics prediction on the target grid cell s_{ij}.

Definition 3 (Urban dynamic features $\boldsymbol{X}$). Features that impose influences on the urban dynamic status are urban dynamic features, *e.g.*, traffic speed, crowd flow, time, *etc*. One feature of a grid cell s at time slot t is a scalar x_t. Features from each grid cell in a target region at time slot t forms the feature map of the target region R, denoted as a matrix $X_t \in \mathbb{R}^{\ell \times \ell}$. Combining features from multiple aspects gives the complete feature maps in region R at time slot t, *i.e.*, a tensor $\boldsymbol{X}_t = \{X_t^1, ..., X_t^n\} \in \mathbb{R}^{n \times \ell \times \ell}$, where $n \in \mathbb{N}^+$ is the number of features.

Definition 4 (Urban status y). Urban status depicts and evaluates the condition of urban mobility, *e.g.*, the traffic quality measured in traffic inflow/outflow. We denote y_t as the urban status of grid cell s in time slot t. In this paper, we choose one urban status measure as the prediction target. Note that other measures can be treated as urban dynamic features to assist the prediction.

Definition 5 (Urban dynamic history $\mathcal{D}$.) Urban dynamics observations $\mathcal{D}$ records urban features and status sequentially over time, *i.e.*, $\mathcal{D} = \{\boldsymbol{X}_t, y_t\}_{t=1}^T$, where T is the length of $\mathcal{D}$. Specifically, we use $\mathcal{D}_t$ to denote the historical observations prior to time slot $(t+1)$ with $\mathcal{D}_t = \{\boldsymbol{X}_1, y_1, \cdots, \boldsymbol{X}_t, y_t\}$.

B. Problem Definition & Technical Challenges

Unlike methods that require task segmentation [13]–[15], we view urban dynamics data as a sequence of continuous observations over time. Specifically, at a time slot t with prior dynamics history $\mathcal{D}_{t-1}$, the current urban features $\boldsymbol{X}_t$ are observed for prediction, after which the ground-truth urban status y_t can be perceived. Therefore, the continuous urban dynamics prediction problem aims at training a model that is able to predict the urban status y_t from the current observation $\boldsymbol{X}_t$ and $\mathcal{D}_{t-1}$ for the target grid cell s in a time slot t. For example, given the urban features and urban dynamic history of a region at step t, including travel demands and weather statistics, it is worth predicting the urban status as the traffic speed, for traffic light control, route recommendation, *etc*. Since the pattern of the urban dynamics (namely, the prediction task) evolves over time with uncertainties, we model the problem using the meta-learning paradigm as below.

Problem Definition (Continuous Urban Dynamics Prediction). Given historical urban dynamics data $\mathcal{D} = \{\boldsymbol{X}_t, y_t\}_{t=1}^T$, we aim to train a meta-learner parameterized by θ that is able to minimize the expected loss towards the end of the dynamics sequence, *i.e.*,

$$\theta^* = \arg\min_{\theta} \sum_{t=1}^{T} \mathcal{L}(\phi_t, \boldsymbol{X}_t, y_t), \text{ where } \phi_t = q(\boldsymbol{X}_t, \mathcal{D}_{t-1}; \theta). \tag{1}$$

Note that, here T represents the end of an urban dynamics time series data. It can take ∞ if the dynamics data keeps being generated over time. The well-trained meta-learner θ^* can then be used to predict urban status for future time slots.

Challenges. As are illustrated in the introduction, the introduced continuous urban dynamics prediction problem is challenging from two perspectives: (**C1**) How to characterize the domain shifts (*i.e.*, urban dynamic pattern variations) and adapt to them in the urban dynamics modeling (See Fig. 1(a))? (**C2**) Given urban dynamics data observed continuously, how to design an algorithm that can adapt to new observations and make predictions sequentially free from the task segmentation assumption (See Fig. 1(b))?

III. Methodologies

Built upon the state-of-the-art (SOTA) approaches in meta-learning (See Section III-A), we propose domain adaptable continuous meta-learning (DAC-ML) for the urban dynamics prediction problem. In DAC-ML, we introduce a domain information inference network (*i.e.*, tackling the challenge **C1**, See Section III-B), and develop a domain adaptable Bayesian meta-learning (DAC-ML) framework to enable continuous adaptation and online prediction given observed and historical urban dynamics data (addressing challenge **C2**, See Section III-C).

A. SOTA: Bayesian Meta-Learning

Traditional meta-learning for few-shot learning with task segmentation [13], [15], [18]–[20] aims at training a model that can adapt to a new task quickly from little support data. For the fulfillment of such a goal, the meta-learner q parameterized by θ is trained on a set of training tasks τ_i's sampled from the same task distribution, *i.e.*, $\tau_i \sim p(\tau)$. Each task τ_i consists of a training set $\mathcal{D}_i^{tr}$ for adaptation and a testing set $\{(\boldsymbol{X}_i^{ts}, y_i^{ts})\}$ for loss computation and meta-learner update. With the negative log likelihood loss, its objective is:

$$\min_{\theta} \mathbb{E}_{\tau_i}[-\log(p(y_i^{ts}|\boldsymbol{X}_i^{ts}, \phi_i))], \text{ where } \phi_i = q(\mathcal{D}_i^{tr}; \theta).$$

Here, the task statistics ϕ_i adapted from $q(\mathcal{D}_i^{tr}; \theta)$ is used for the prediction model. To capture the potential uncertainties of the task statistics, *e.g.*, weather conditions and car accidents for urban dynamics prediction, Bayesian meta-learning [16], [21], [22] infers a distribution of task statistics $q(\phi_i|\mathcal{D}_i^{tr}; \theta)$ for the final prediction rather than a deterministic task statistic. With this inferred distribution, Bayesian meta-learning targets maximizing the log likelihood lower bound[2] across all tasks, *i.e.*,

$$\begin{aligned} \max_{\theta} \mathbb{E}_{\tau_i} \big[& \mathbb{E}_{q(\phi_i|\mathcal{D}_i^{tr};\theta)}[\log p(y_i^{ts}|\boldsymbol{X}_i^{ts}, \phi_i)] \\ & - D_{KL}(q(\phi_i|\mathcal{D}_i^{tr}; \theta)||p(\phi_i; \theta)) \big], \end{aligned} \tag{2}$$

where D_{KL} is the Kullback-Leibler divergence. For simplicity, the Bayesian prior $p(\phi_i; \theta)$ is assumed to be the standard Gaussian distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Here, q is modeled with a task inference network parameterizing the mean and log-variance diagonal of a Gaussian distribution of ϕ_i. Hence,

[2]The log likelihood is approximated by variational lower bound (ELBO). See more details in [16], [21], [22].

ϕ_i can be sampled from this distribution, and the sampling process enables the Bayesian meta-learner to capture the task uncertainty during each adaptation process.

Limitations. Such Bayesian meta-learning paradigm fails to address (**C1**) the domain adaptation challenge and (**C2**) the unsegmented task challenge in the problem of urban dynamics prediction. First, the task distribution $p(\tau)$ may also change across tasks. As illustrated in Fig. 1(a), each month represents a new and unique task (data pattern) distribution. Moreover, the dataset of each task $\mathcal{D}_i^{tr}$ may not be well segmented because the time interval length of data with the same pattern may be time-varying and uncertain (See Fig. 1(b)).

B. Domain Adaptable Bayesian Meta-Learning

Each task τ_i is sampled from a distribution determined by its domain, *e.g.*, the unique urban dynamics pattern in a month. Denote ω_i as the domain statistics of a task τ_i, indicating the data generation pattern of the underlying domain. To address challenge **C1**, we develop *domain adaptable Bayesian meta-learning* (DAC-ML), by introducing domain statistics ω_i's into the objective of single domain Bayesian meta-learning in Eq. (2). For each training task τ_i, the objective of in Eq. (2) is extended as follows.

$$
\begin{aligned}
&\mathbb{E}_{q(\omega_i,\phi_i|\mathcal{D}_i^{tr};\theta)}[\log p(y_i^{ts}|\boldsymbol{X}_i^{ts},\omega_i,\phi_i)] \\
&\quad - \mathbb{E}_{q(\omega_i,\phi_i|\mathcal{D}_i^{tr};\theta)}\left[\log \frac{q(\omega_i,\phi_i|\mathcal{D}_i^{tr};\theta)}{p(\omega_i,\phi_i;\theta)}\right] \\
=&\mathbb{E}_{h(\omega_i|\mathcal{D}_i^{tr};\theta)}\mathbb{E}_{q(\phi_i|\mathcal{D}_i^{tr},\omega_i;\theta)}[\log p(y_i^{ts}|\boldsymbol{X}_i^{ts},\omega_i,\phi_i)] \\
&\quad - \mathbb{E}_{h(\omega_i|\mathcal{D}_i^{tr};\theta)}\mathbb{E}_{q(\phi_i|\mathcal{D}_i^{tr},\omega_i;\theta)}\left[\log \tfrac{h(\omega_i|\mathcal{D}_i^{tr};\theta)q(\phi_i|\mathcal{D}_i^{tr},\omega_i;\theta)}{p(\omega_i;\theta)p(\phi_i|\omega_i;\theta)}\right] \\
=&\mathbb{E}_{h(\omega_i|\mathcal{D}_i^{tr};\theta)}\mathbb{E}_{q(\phi_i|\mathcal{D}_i^{tr},\omega_i;\theta)}[\log p(y_i^{ts}|\boldsymbol{X}_i^{ts},\omega_i,\phi_i)] \\
&\quad - D_{KL}\big[h(\omega_i|\mathcal{D}_i^{tr};\theta)||p(\omega_i;\theta)\big] \\
&\quad - \mathbb{E}_{h(\omega_i|\mathcal{D}_i^{tr};\theta)}D_{KL}\big[q(\phi_i|\mathcal{D}_i^{tr},\omega_i;\theta)||p(\phi_i|\omega_i;\theta)\big],
\end{aligned} \tag{3}
$$

where the first equality holds by the definition of conditional probability. Taking the average over all training tasks $\tau_i \sim p(\tau)$, **the DAC-ML objective** is Eq. (4).

$$
\begin{aligned}
\max_{\theta}\mathbb{E}_{\tau_i}\Big[&\mathbb{E}_{h(\omega_i|\mathcal{D}_i^{tr};\theta)}\mathbb{E}_{q(\phi_i|\omega_i,\mathcal{D}_i^{tr};\theta)}\big[\log p(y_i^{ts}|\boldsymbol{X}_i^{ts},\omega_i,\phi_i)\big] \\
&- D_{KL}\big(h(\omega_i|\mathcal{D}_i^{tr};\theta)||p(\omega_i;\theta)\big) \\
&- \mathbb{E}_{h(\omega_i|\mathcal{D}_i^{tr};\theta)}\big[D_{KL}\big(q(\phi_i|\omega_i,\mathcal{D}_i^{tr};\theta)||p(\phi_i|\omega_i;\theta)\big)\big]\Big].
\end{aligned} \tag{4}
$$

C. Continuous Domain Adaptable Bayesian Meta-Learning

For the second challenge **C2**, considering that urban dynamics data come sequentially, and at each time slot t, the current urban dynamic features $\boldsymbol{X}_t$ are observed, and we are able to derive the probability of the current urban status as $p(y_t|\boldsymbol{X}_t,\mathcal{D}_{t-1})$. To capture such a continuity, we treat previous observations $\mathcal{D}_{t-1}$ and current observations $\boldsymbol{X}_t$ as the training dataset $\mathcal{D}_t^{tr} = (\mathcal{D}_{t-1}^{tr}, \boldsymbol{X}_t)$ to adapt and infer task and domain statistics from, and the current observations $\boldsymbol{X}_t$ as the test sample. Clearly, over time t, the dataset of the current task is with a variable length. Then, replacing the expectation $\mathbb{E}_{\tau_i}$ over the distribution of tasks in Eq. (4) with a summation over the time t, we extend the domain adaptable meta-learning to fit the setting of continuous prediction, as the objective of domain adaptable continuous meta-learning (DAC-ML), *i.e.*,

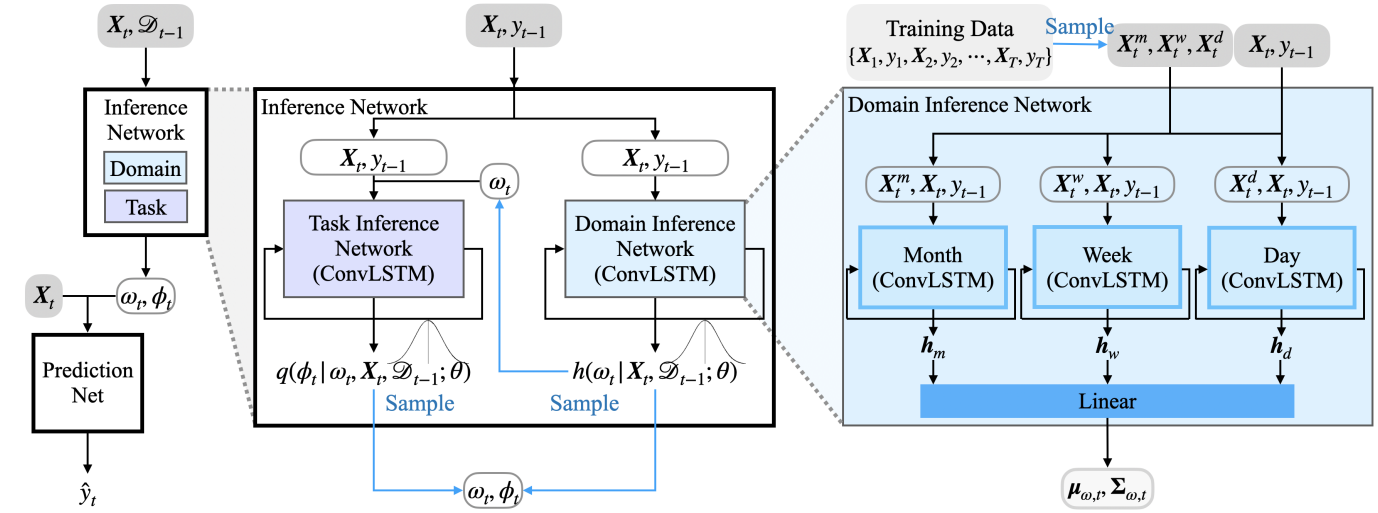
$$
\begin{aligned}
\max_{\theta}\sum_{t=1}^{T}\Big[&\mathbb{E}_{h(\omega_t|\boldsymbol{X}_t,\mathcal{D}_{t-1};\theta)}\mathbb{E}_{q(\phi_t|\omega_t,\boldsymbol{X}_t,\mathcal{D}_{t-1};\theta)}\big[\log p(y_t|\boldsymbol{X}_t,\mathcal{D}_{t-1},\omega_t,\phi_t)\big] \\
&- D_{KL}\big(h(\omega_t|\boldsymbol{X}_t,\mathcal{D}_{t-1};\theta)||p(\omega_t;\theta)\big) \\
&- \mathbb{E}_{h(\omega_t|\boldsymbol{X}_t,\mathcal{D}_{t-1};\theta)}\big[D_{KL}\big(q(\phi_t|\omega_t,\boldsymbol{X}_t,\mathcal{D}_{t-1};\theta)||p(\phi_t|\omega_t;\theta)\big)\big]\Big],
\end{aligned} \tag{5}
$$

where $p(\omega_t;\theta)$ and $p(\phi_t|\omega_t;\theta) \sim \mathcal{N}(\mathbf{0},\boldsymbol{I})$. In particular, Eq. (5) does not require us to know a priori the distribution $p(\tau)$ of tasks, as the distribution $q(\phi_t|\omega_t,\boldsymbol{X}_t,\mathcal{D}_{t-1};\theta)$ of task statistics ϕ_t is inferred at each time t.

We model the distributions of task statistics ϕ and domain information ω using two neural networks, *i.e.*, q and h, respectively. Therefore, the goal of DAC-ML is to jointly meta-learn both the task and domain inference networks such that they can learn representations of the data collected so far. With this, we show the overall structure of DAC-ML in Fig. 2(a). DAC-ML is composed of two networks, namely, the *inference net* and the *prediction net*. The inference network is responsible for learning both task and domain statistics ϕ_t and ω_t given observations, and the prediction net makes the prediction of $\hat{y}_t$ based on the current observation $\boldsymbol{X}_t$ and the learned domain and task statistics. In order to capture both domain and task statistics, the inference network (See Fig. 2(b)) contains two modules, *i.e.*, the domain inference network and the task inference network, which capture domain and task statistics, respectively. We then introduce the DAC-ML algorithms for meta-training and meta-testing. Their pseudo-codes are presented in Alg. 1 and Alg. 2 respectively.

Meta-training procedure. As is shown in Alg. 1, for each iteration in training, we sample a sequence of urban dynamics data from the training data $(\boldsymbol{X}_{1:T}, y_{1:T})$ to train and update the meta-learner θ (Line 2). Following the continuous setting, we observe the current urban dynamic features $\boldsymbol{X}_t$ and aim to infer the task statistics and domain statistics distributions at time step t. We sample one domain statistics ω_t from the domain inference network, and one task statistics ϕ_t from the task inference network (Line 4-6). With sampled task statistics ϕ_t and domain information ω_t at time step t, a prediction $\hat{y}_t$ is made (Line 7). Then, we observe ground-truth urban status y_t, with which the negative log loss can be calculated (Line 8). We also update historical observations up to time step t for further task and domain inference (Line 9). After finishing loss calculation of a complete sequence, we update the meta-learner θ with Eq. (5) (Line 11).

Meta-testing procedure. After training, the well-trained meta-learner θ is expected to quickly adapt to a new sequence via Alg. 2. At the beginning of the testing procedure, only one initial urban dynamics feature $\boldsymbol{X}_1$ is observed (Line 2). Same with other time steps t, we infer task statistics ϕ_t and domain information ω_t from the observations (Line 3-4). The inferred statistics are used to conduct predictions (Line 5) until we



(a) DAC-ML structure. (b) Inference network architecture. (c) Disentangled domain inference network.

Fig. 2: Our DAC-ML architecture.

Algorithm 1 DAC-ML Meta-Training

Require: Training data $\boldsymbol{X}_{1:T}, y_{1:T}$, number of training iterations N, initial model parameters θ.

1: **for** $i = 1, \cdots N$ **do**
2: Sample training batch $\boldsymbol{X}_{1:n}, y_{1:n}$ from the full time series.
3: **for** $t = 1 \cdots n$ **do**
4: Observe $\boldsymbol{X}_t$.
5: Infer $h(\omega_t|\boldsymbol{X}_t, \mathcal{D}_{t-1}; \theta)$ and sample one ω_t.
6: Infer $q(\phi_t|\omega_t, \boldsymbol{X}_t, \mathcal{D}_{t-1}; \theta)$ and sample one ϕ_t.
7: Predict $p_\theta(\hat{y}_t|\boldsymbol{X}_t, \mathcal{D}_{t-1}, \omega_t, \phi_t)$.
8: Observe y_t and incur NLL loss $\ell_t = -\log p_\theta(\hat{y}_t|\boldsymbol{X}_t, \mathcal{D}_{t-1}, \omega_t, \phi_t)$.
9: Update $\mathcal{D}_t = \{\boldsymbol{X}_1, y_1, \cdots \boldsymbol{X}_t, y_t\}$.
10: **end for**
11: Update θ based on Eq. (5).
12: **end for**

Algorithm 2 DAC-ML Meta-Testing

Require: Testing data $\boldsymbol{X}_1$, $\mathcal{D}_0 = \emptyset$, well-trained DAC-ML parameters θ.

1: **for** $t = 1 \cdots T$ **do**
2: Observe $\boldsymbol{X}_t$.
3: Infer $h(\omega_t|\boldsymbol{X}_t, \mathcal{D}_{t-1}; \theta)$ and sample one ω_t.
4: Infer $q(\phi_t|\omega_t, \boldsymbol{X}_t, \mathcal{D}_{t-1}; \theta)$ and sample one ϕ_t.
5: Predict $p_\theta(\hat{y}_t|\boldsymbol{X}_t, \mathcal{D}_{t-1}, \omega_t, \phi_t)$.
6: Observe y_t, and update $\mathcal{D}_t = \{\boldsymbol{X}_1, y_1, \cdots \boldsymbol{X}_t, y_t\}$.
7: **end for**

observe the true urban status and update the historical urban dynamic features for future prediction (Line 6).

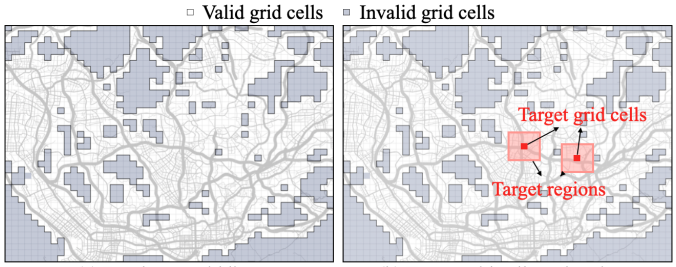
D. Practical Challenges of DAC-ML

Given the DAC-ML structure and algorithms, we will tackle two practical challenges for continuous urban dynamics prediction problem, **P1**) How to implement the task and domain inference networks to capture the task and domain statistics from variable length historical data? **P2**) How to characterize and represent the domains of the continuous urban dynamics prediction problem?

Solution to **P1**. From the urban dynamics data, we model both the domain and the task inference networks using ConvLSTMs [9] as shown in Fig. 2(b). Since the ground truth urban status is only observed after a prediction is made, we employ a one-step time difference approach and input a previous step ground truth urban status y_{t-1} with the current observation $\boldsymbol{X}_t$ into the inference network. For the domain inference network, we view the hidden state output $\boldsymbol{h}_{\omega,t-1}$ at a previous time step $t-1$ as an embedding of the domain statistics from historical data $(\mathcal{D}_{t-2}, \boldsymbol{X}_{t-1})$. Then, at each time step t, we observe the current urban dynamic features $\boldsymbol{X}_t$ and previous urban status y_{t-1}, and input it with $\boldsymbol{h}_{\omega,t-1}$ to the domain inference network to generate the current hidden state $\boldsymbol{h}_{\omega,t}$. $\boldsymbol{h}_{\omega,t}$ is then fed as the input to a linear layer to derive the mean and log variance for the Gaussian distribution of domain statistics $h(\omega_t|\boldsymbol{X}_t, \mathcal{D}_{t-1}; \theta)$. The hidden state $\boldsymbol{h}_{\omega,t}$ also serves as the input to the next step domain statistics inference. A domain statistics ω_t is sampled according to such distribution. Similarly, the task inference network works the same way except that we input the task inference network with an additional domain statistics ω_t sampled from its distribution $h(\omega_t|\boldsymbol{X}_t, \mathcal{D}_{t-1}; \theta)$. This fulfills the dependency of task statistics on the domain information.

Considering that $\boldsymbol{X}_t$ is a tensor for each time slot t recording urban dynamic features of surrounding ℓ grid cells, we use ConvLSTM to emphasize the spatial correlations among locations rather than the LSTM network.

Solution to **P2**. Fig. 1 illustrates that the urban dynamics patterns across months show significant changes, which is one crucial dimension of the underlying domain. As observed in [23], [24] the urban dynamics patterns shift across months,



(a) Road map gridding. (b) Target grid cells and regions.

Fig. 3: Map gridding and target grid cells and regions.

weeks, and days. Hence, we model the domain inference network with three independent ConvLSTM components as is shown in Fig. 2(c). Here, each component models temporal uncertainties from one temporal granularity. Fig. 2(c) shows three such granularities, namely, monthly, weekly and daily.

To train and distinguish the month, week, and day domain inference components, we feed them with different data in addition to the current observations $\boldsymbol{X}_t$ and y_{t-1}. Specifically, we follow [24], and sample historical observations that share monthly, weekly, and daily information to the month, week, and day inference components, respectively. At time slot t, the sampled historical observations contain samples $[\boldsymbol{X}_{t-\lfloor \frac{t}{p} \rfloor \cdot p}, \boldsymbol{X}_{t-(\lfloor \frac{t}{p} \rfloor -1)\cdot p}, \cdots, \boldsymbol{X}_{t-p}]$, where p is a fixed period, *i.e.*, one month, week, day. We average the sampled historical observations, $\boldsymbol{X}_t = \frac{1}{\lfloor \frac{t}{p} \rfloor} \sum_{l=1}^{\lfloor \frac{t}{p} \rfloor} \boldsymbol{X}_{t-l\cdot p}$, to construct the input for domain inference net at different granularities.

IV. Experiments

In this section, we evaluate the performance of the DAC-ML framework using three real-world urban dynamics datasets, including (1) traffic speed, (2) vehicle inflow, and (3) travel demand. These datasets were collected from traffic data in Shenzhen, China, from July 1st to December 31st in 2016. Our experimental results on these datasets show that *i)* DAC-ML outperforms all baselines in predicting urban dynamics along the timeline (See Section IV-C); *ii)* the domain information disentangled and extracted from the domain inference networks (in our DAC-ML model) successfully captures the domain shifts in urban dynamics (See Section IV-D); and *iii)* our DAC-ML model is robust to changes of various hyper-parameters (See Section IV-E).

A. Data Descriptions and Preparation

Dataset description. Using the map data from OpenStreetMap [25], we partition the Shenzhen City in China into $1km \times 1km$ grid cells, resulting in a total of 1,934 valid grid cells. For each target grid cell, we consider its target region as the 5×5 surrounding grid cells centering at the target grid cell. This results in a total of $1,656$ possible target grid cells, as illustrated in Fig. 3.

We extract the *traffic speeds*, *taxi inflows* and *travel demands* from taxi GPS records collected in Shenzhen, China, from July 1st to December 31st in 2016. We define each time slot t as an hour-long period. Then, the taxi inflow of a time slot t is estimated from the number of taxis that stay or arrive at a target grid cell. The traffic speed is calculated over the average speed of each taxi coming into a target grid cell in time slot t. The travel demand in a target grid cell is the total number of taxi pickups during time slot t.

Data preparation. We prepare our urban data for two urban dynamics prediction problems in our experiments, including the continuous predictions for *traffic speed* and *taxi inflow*, respectively. Below, we detail the two problems with their supporting urban dynamics features.

- Traffic speed prediction. In speed prediction, the target urban dynamic status in each grid cell measures the average traffic speed. A total of roughly $2,000$ time slots are available in the time series data, where the first 80% of the sequence is used for training a good meta-learner, and the remaining 20% is used for testing and evaluation. In this prediction, travel demands, traffic inflow, and the time of the day are treated as urban dynamics-related features. The objective of this task is to predict the traffic speed of a target grid cell s based on the available historical features and observations.
- Taxi inflow prediction. Similar to traffic speed prediction, in the taxi inflow prediction, taxi inflow is the urban dynamic status of interest in each grid cell s. In this instance, travel demands, traffic speed, and the time of day are urban dynamics-related features. We also use a time series with roughly $2,000$ time slots for the experiment, where the first 80% portion is used for meta-training and the remaining 20% is used for meta-testing and evaluating.

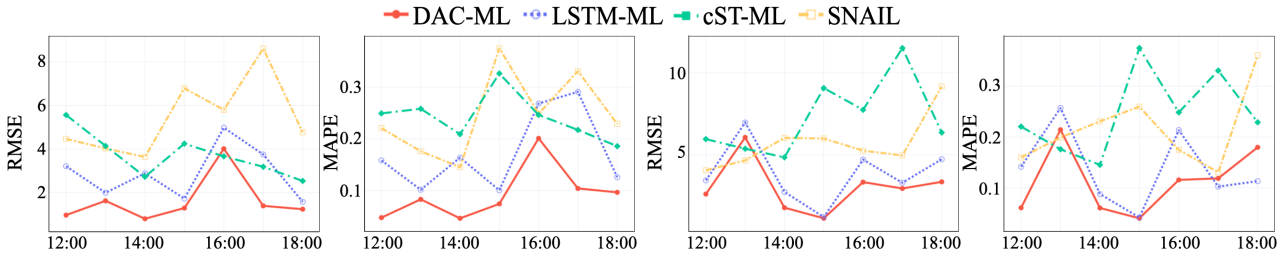
B. Experimental Settings

In this section, we detail the experimental settings, including the baseline methods and evaluation metrics. We delegate more experiment details in Appendix A.

Baselines. We compare DAC-ML with three baselines below:

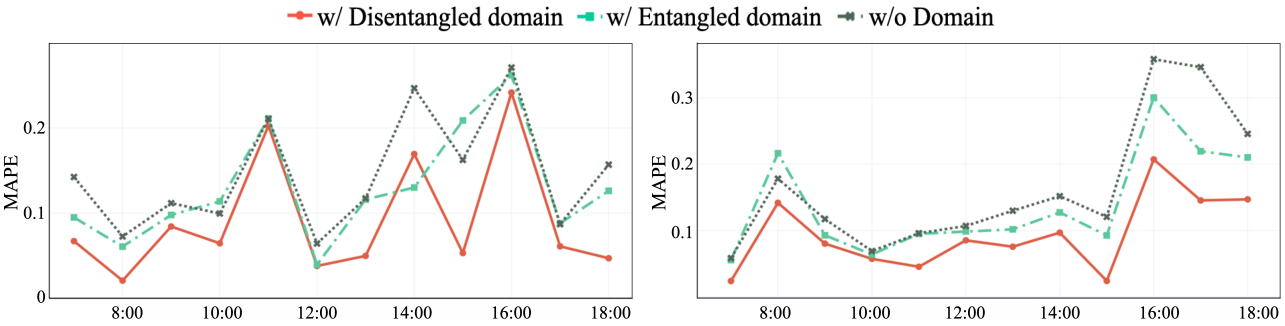
- **LSTM-ML [26]** stands for the LSTM meta-learner. It uses the hidden state of an LSTM [27] to encode the current sample $\boldsymbol{X}_t$, the label of the previous sample y_{t-1}, and the history embedding $\boldsymbol{h}_{t-1}$ into a new embedding $\boldsymbol{h}_t$ for the next prediction. This model naturally characterizes the temporal dependency across data points along time.
- **SNAIL [18]** is a state-of-the-art black-box meta-learning approach which does not capture model uncertainty. It assumes task separation (as a time window of a day in our experiments), and applies the attention layers to extract task statistics for prediction.
- **cST-ML [15]** is a state-of-the-art black-box Bayesian meta-learning approach that captures task uncertainties for time series data with clear task separation (as a time window of a day in our experiments).

When comparing with baselines, we use the same set of training and testing data. For methods that require task-segmentation and supporting data, we follow their original setups to use one day's urban dynamics as a task for both training and testing. For a fair comparison with such baselines, we use the first five time slots' urban dynamics data to enable task adaptation for all compared methods.



(a) RMSE on traffic speed. (b) MAPE on traffic speed. (c) RMSE on taxi inflow. (d) MAPE on taxi inflow.

Fig. 4: Performance comparison in 7 consecutive hours on traffic speed and taxi inflow prediction.



(a) MAPE in 12 consecutive hours on traffic speed prediction. (b) MAPE in 12 consecutive hours on taxi inflow prediction.

Fig. 5: Domain information influence on traffic speed and taxi inflow predictions in 12 consecutive hours.

Evaluation metrics. We use mean absolute percentage error (MAPE), *i.e.*, $\text{MAPE} = \frac{1}{T}\sum_{t=1}^{T}\frac{|y_t-\hat{y}_t|}{y_t}$, and rooted mean square error (RMSE), *i.e.*, $\text{RMSE} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_t-\hat{y}_t)^2}$ for evaluations. Here, y_t denotes the ground-truth urban status value observed in the target grid cell s at the t-th time slot, and $\hat{y}_t$ is its related prediction. We use T to denote the time span or the total number of time slots to conduct continuous urban dynamics prediction on.

C. DAC-ML Performance

Fig. 4 summarizes the performance of our proposed DAC-ML in comparison to three baseline methods in predicting traffic speed and taxi inflow given historical urban dynamics data. We predict the traffic speeds and taxi inflows in four randomly selected target grid cells over five days for each approach. To avoid randomness, the results shown in Fig. 4 are averaged across the four grid cells and five days. We report the prediction errors of urban dynamics in RMSE and MAPE.

From Fig. 4, our proposed DAC-ML outperforms baselines with the lowest overall prediction errors in RMSE and MAPE. This is mainly because of the well extracted and disentangled domain information and a continuous training scheme in DAC-ML, which enables a longer term task and domain information memorization and generalization. Moreover, the two baselines, cST-ML and SNAIL, have in general the worst results, comparing to both our DAC-ML and LSTM-ML. This is because both cST-ML and SNAIL predefine the length of the prediction task (as 5 hours in our experiment) thus ignoring the uncertainty of the task length, while DAC-ML and LSTM-ML use recurrent neural networks to embed and consider the long-term historical data in the prediction. In detail, DAC-ML outperforms all the baselines on both datasets, and improves the best performance of the baselines by up to 36.2% in RMSE and 30.6% of MAPE. Next, we evaluate how the domain information extracted in DAC-ML contributes to the urban dynamics prediction problem.

D. Impact of Extracted Domain Information

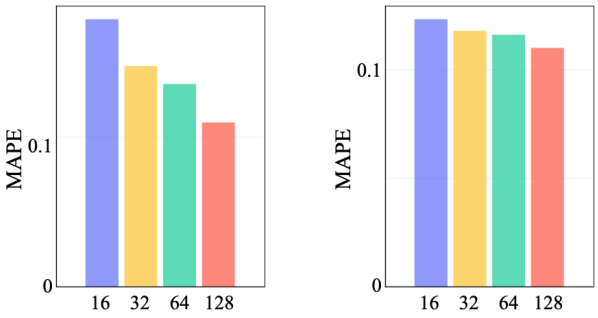
The domain information extracted in DAC-ML contributes significantly to the urban dynamics prediction problem. In this section, we compare alternative designs in the domain inference network for domain information extraction, including the proposed DAC-ML with disentangled domain, DAC-ML with entangled domain, and DAC-ML without domain. DAC-ML with entangled domain uses one domain inference network to extract a unifying domain representation, DAC-ML without domain does not have a domain inference network. In Fig. 5, we observe that DAC-ML with disentangled domain always outperforms the other two design alternatives. First, DAC-ML without domain fails to capture the domain shift over time, thus performs the worst. Moreover, when comparing both versions of DAC-ML that include a domain inference network, DAC-ML with disentangled domain performs better than that with entangled domain. This is because DAC-ML with disentangled domain extracts multiple independent domain dimensions – daily, weekly, and monthly – thus avoiding overfitting in the high-dimensional domain space [28], [29].

E. Ablation Studies

In this section, we investigate how the performance of our proposed DAC-ML changes, with i) different prediction horizons and ii) different sizes of the learned hidden states.

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

Methods	Traffic Speed				Vehicle Inflow			
	1h		3h		1h		3h	
	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE
SNAIL	5.442	0.247	4.342	0.288	5.725	0.216	5.876	0.276
cST-ML	3.719	0.241	5.604	0.361	7.255	0.246	6.367	0.333
LSTM-ML	2.871	0.201	3.754	0.243	3.771	0.130	6.230	0.282
DAC-ML	**1.717**	**0.127**	**3.022**	**0.177**	**3.352**	**0.126**	**4.317**	**0.268**



(a) Hidden state $\boldsymbol{h}$ dimensions. (b) $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ dimensions.

Fig. 6: Performance in MAPE with different hyper-parameters on traffic speed prediction.

The impact of the prediction horizon. Now, we further examine the prediction performances on the urban dynamics, with different prediction horizons. In the experiments, we compare the results of the different approaches for prediction horizons of one hour and of three hours. In Table I, we observe that in both scenarios, our DAC-ML consistently outperforms the three baselines. Moreover, unsurprisingly, for all the approaches, the prediction for the 1-hour horizon has a lower error than for the 3-hour horizon.

The impact of the hidden state size. For the hyper-parameter analysis, we investigate how the size of hidden states $\boldsymbol{h}$ in ConvLSTMs (inside both the task and domain inference networks of DAC-ML) and the size of mean $\boldsymbol{\mu}$ and log variance $\boldsymbol{\Sigma}$ impact the model performance. Their results in Fig. 6a and Fig. 6b show the prediction errors of traffic speed in RMSE and MAPE, with different sizes of the hidden states. We observe that the model performance is sensitive to both hyper-parameters. Higher dimensions of these hidden states lead to better performance. This is because the larger hidden states can potentially capture more information of the underlying domain and task. We omit the results of taxi inflow prediction for brevity since they share the same pattern as the traffic speed prediction.

V. Related Work

In this section, we summarize the literature from three related areas: urban dynamics prediction, meta-learning, and continuous learning.

Urban dynamics prediction aims to predict the future urban status, such as traffic volume, crowd flow, *etc*. Previous works primarily apply traditional machine learning approaches [3]–[6], [30], or employ deep neural networks [7]–[12], [31], [32] to model the spatial and temporal correlations in urban data. These methods focus on traffic prediction, and use historical traffic data to capture correlations among the past traffic, surrounding environment features and the future traffic. Specifically, ConvLSTM [9] is used for traffic accident prediction and crowd density estimation [7], [8]. Cui *et al*. [10] and Yu *et al*. [11] apply a combination of CNN and LSTM to predict traffic speed. Yao *et al*. [31] proposes a deep multi-view spatial-temporal network (DMVST-Net) to model temporal and spatial correlations, and semantic information among regions with similar temporal patterns for taxi demand prediction. Stacked autoencoders are used for travel demand prediction in [12]. Some recent works target the uncertainties in urban dynamics and strive to use meta-learning approaches for urban dynamics prediction [13]–[15]. To capture the spatial-temporal correlations in urban dynamics, Pan *et al*. [13] attempts to use a meta-graph attention network to extract spatial correlation, and a meta-recurrent neural network for temporal dependency for traffic prediction. Yao *et al*. [14] takes another perspective, and views different cities as different training tasks, and proposes to meta-learn knowledge across multiple cities, which is then leveraged to predict traffic status. The cST-ML [15] method covers urban uncertainties with Bayesian meta-learning [16]. However, these works tend to consider the urban dynamic uncertainties from one unified domain. In contrast to previous works, our work models such uncertainties by modeling the urban dynamics with high-dimensional domain shifts, and estimates these uncertainties by developing a domain adaptable Bayesian meta-learning framework.

Meta-learning. Meta-learning aims to learn a model from training tasks by rapidly adapting to a novel task when only a small number of test samples are available. There are three lines of work tackling the meta-learning problem, specifically, optimization-based [19], [33], model-based [18], [34], and metrics-based [35]–[37] meta-learning algorithms. Optimization-based models [19], [33] aim to learn from training tasks a good initialization so that it forms an appropriate inductive bias for fast adaptation via gradient descent. Model-based meta-learning methods [18], [34] capture task statistics from a support dataset and output the adapted parameters for prediction. Metrics-based meta-learning approaches [35]–[37] endeavor to learn a distance function for comparing two different samples from different tasks. Moreover, some recent works focus on better capturing task uncertainty and heterogeneity (*e.g*., [16], [21], [22], [38], [39]) via maintaining a knowledge base to capture, store and generalize task-related information for quick adaptation [38], [39]. For example, Yao *et al*. [38] models knowledge base as a tree, and [39] as a graph. Finally, many works [16], [21], [22] propose Bayesian meta-learning approaches, which, rather than learning deterministic task statistics, capture the distribution of the task statistics. For example, MOCA [17] utilizes a differentiable Bayesian changepoint detection scheme to detect potential task changes in a sequence. In contrast to the state-of-the-art approaches, which do not consider the uncertainties of higher-dimensional task domains, our approach attempts to mitigate this problem by disentangling the domain. In addition, we focus on urban

uncertainties from spatial-temporal data.

Continuous learning. Continuous learning corresponds to the problem to learn from a streaming series of tasks and reuses the learned information for current task prediction [40]–[42]. Most methods [40], [43]–[46] use regularization to avoid forgetting in continual learning or put a limitation on parameter update given a new task. Unlike these approaches, our DAC-ML explicitly learns a prior over task and domain distributions, automatically avoiding negative transfer by detecting task and domain shifts and adapting to new test samples.

VI. Discussions & Conclusions

In this paper, we solve the continuous urban dynamics prediction problem from multiple domains that do not require task segmentation. A novel domain-adaptable continuous meta-learner (DAC-ML) is proposed, which advances Bayesian meta-learning towards a continuous setting. The proposed DAC-ML can learn a general urban dynamics prediction strategy via extracting and disentangling task and domain statistics from historical urban dynamics data, and capable of quick adaptation to new observations in an online manner, leading to improved performance in urban dynamics prediction problems. Novel training and testing algorithms are designed for DAC-ML, where task and domain information can be embedded and captured continuously by ConvLSTM networks. Extensive experiments on real-world traffic datasets (*i.e.*, traffic speed, taxi inflow and travel demand) are conducted to evaluate our proposed DAC-ML method. Our experimental results demonstrate the facts that i) DAC-ML is compatible to the continuous prediction setting and thus outperforms baselines requiring task segmentation, ii) DAC-ML captured domain statistics contribute to a more precise prediction, and iii) DAC-ML's disentanglement strategy is an efficient approach to utilizing domain information, leading to superior performance in urban dynamics prediction compared to previous approaches.

Acknowledgment

Xin Zhang and Yanhua Li were supported in part by NSF grants IIS-1942680 (CAREER), CNS-1952085, CMMI-1831140, and DGE-2021871. Xun Zhou was 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. Ziming Zhang was supported in part by NSF CCF-2006738. Jun Luo was partially supported by ARC Discovery Project (grant# DB210100743). Oren Mangoubi and Vincent Filardi were supported in part by NSF grant CCF-2104528.

References

[1] A. M. Nagy and V. Simon, "Survey on traffic prediction in smart cities," *Pervasive and Mobile Computing*, vol. 50, pp. 148–163, 2018.

[2] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: concepts, methodologies, and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 5, no. 3, pp. 1–55, 2014.

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

[4] 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*, vol. 137, pp. 59–68, 2016.

[5] Y. Sun, B. Leng, and W. Guan, "A novel wavelet-svm short-time passenger flow prediction in beijing subway system," *Neurocomputing*, vol. 166, pp. 109–121, 2015.

[6] X. Luo, L. Niu, and S. Zhang, "An algorithm for traffic flow prediction based on improved sarima and ga," *KSCE Journal of Civil Engineering*, vol. 22, no. 10, pp. 4107–4115, 2018.

[7] Z. Yuan, X. Zhou, and T. Yang, "Hetero-convlstm: A deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 984–992, 2018.

[8] A. Zonoozi, J.-j. Kim, X.-L. Li, and G. Cong, "Periodic-crn: A convolutional recurrent model for crowd density prediction with recurring periodic patterns.," in *IJCAI*, pp. 3732–3738, 2018.

[9] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," *arXiv preprint arXiv:1506.04214*, 2015.

[10] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction," *arXiv preprint arXiv:1801.02143*, 2018.

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

[12] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2014.

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

[14] 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*, pp. 2181–2191, 2019.

[15] Y. Zhang, Y. Li, X. Zhou, and J. Luo, "cst-ml: Continuous spatial-temporal meta-learning for traffic dynamics prediction," in *International Conference on Data Mining*, 2020.

[16] C. Finn, "Bayesian meta-learning," *URL https://cs330.stanford.edu/slides/ cs330 bayesian metalearning.pdf.[Online]*, 2019.

[17] J. Harrison, A. Sharma, C. Finn, and M. Pavone, "Continuous meta-learning without tasks," *arXiv preprint arXiv:1912.08866*, 2019.

[18] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel, "A simple neural attentive meta-learner," *arXiv preprint arXiv:1707.03141*, 2017.

[19] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," *arXiv preprint arXiv:1703.03400*, 2017.

[20] M. Zhang, H. Marklund, A. Gupta, S. Levine, and C. Finn, "Adaptive risk minimization: A meta-learning approach for tackling group shift," *arXiv preprint arXiv:2007.02931*, 2020.

[21] J. Yoon, T. Kim, O. Dia, S. Kim, Y. Bengio, and S. Ahn, "Bayesian model-agnostic meta-learning," in *Advances in Neural Information Processing Systems*, pp. 7332–7342, 2018.

[22] S. Ravi and A. Beatson, "Amortized bayesian meta-learning," in *International Conference on Learning Representations*, 2018.

[23] X. Yi, J. Zhang, Z. Wang, T. Li, and Y. Zheng, "Deep distributed fusion network for air quality prediction," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 965–973, 2018.

[24] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi, "Dnn-based prediction model for spatio-temporal data," in *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 1–4, 2016.

[25] OpenStreetMap, "Road map data," 2016. data retrieved from Open Street Map, http://www.openstreetmap.org/.

[26] S. Hochreiter, A. S. Younger, and P. R. Conwell, "Learning to learn using gradient descent," in *International Conference on Artificial Neural Networks*, pp. 87–94, Springer, 2001.

[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[28] K. KOUTROUMBAS and S. THEODORIDIS, "Pattern recognition second edition," 2018.
[29] G. Houghes, "On the mean accuracy of statistical pattern recognition," *IEEE Trans. Inform. Theory*, vol. 14, no. 1, pp. 55–63, 1968.
[30] X. Zhan, Y. Zheng, X. Yi, and S. V. Ukkusuri, "Citywide traffic volume estimation using trajectory data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 2, pp. 272–285, 2016.
[31] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.
[32] X. Zhang, C. Huang, Y. Xu, L. Xia, P. Dai, L. Bo, J. Zhang, and Y. Zheng, "Traffic flow forecasting with spatial-temporal graph diffusion network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 15008–15015, 2021.
[33] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," 2016.
[34] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *International conference on machine learning*, pp. 1842–1850, 2016.
[35] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, *et al.*, "Matching networks for one shot learning," in *Advances in neural information processing systems*, pp. 3630–3638, 2016.
[36] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2, Lille, 2015.
[37] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," 2017.
[38] H. Yao, Y. Wei, J. Huang, and Z. Li, "Hierarchically structured meta-learning," in *International Conference on Machine Learning*, pp. 7045–7054, PMLR, 2019.
[39] H. Yao, X. Wu, Z. Tao, Y. Li, B. Ding, R. Li, and Z. Li, "Automated relational meta-learning," in *International Conference on Learning Representations*, 2019.
[40] E. Hazan, "Introduction to online convex optimization," *arXiv preprint arXiv:1909.05207*, 2019.
[41] Z. Chen and B. Liu, "Lifelong machine learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 12, no. 3, pp. 1–207, 2018.
[42] R. Aljundi, K. Kelchtermans, and T. Tuytelaars, "Task-free continual learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11254–11263, 2019.
[43] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.
[44] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Learning to generalize: Meta-learning for domain generalization," *arXiv preprint arXiv:1710.03463*, 2017.
[45] M. Riemer, I. Cases, R. Ajemian, M. Liu, I. Rish, Y. Tu, and G. Tesauro, "Learning to learn without forgetting by maximizing transfer and minimizing interference," *arXiv preprint arXiv:1810.11910*, 2018.
[46] J. Xu and Z. Zhu, "Reinforced continual learning," *arXiv preprint arXiv:1805.12369*, 2018.

APPENDIX

A. Implementation Details

The reported RMSE and MAPE are averaged over four randomly selected target grid cells over five days. After each training epoch, we evaluate the average, best, and worst RMSE and MAPE for DAC-ML and all baselines. Comparing the average MAPE score during training, we choose the best models for DAC-ML and all baselines. The best models are then compared with their results presented. To support the reproducibility of the results in this paper, we detail both the DAC-ML and baseline implementation settings. All models are trained using Adam optimizer with $\beta_1 = 0.5$ and $\beta_2 = 0.999$, and a learning rate of 2^{-4} for 1,000 epochs.

- **LSTM-ML** is composed of a task inference network and a posterior prediction net. For a fair comparison with the proposed DAC-ML, the task inference network is implemented with a layer of CNN following a layer of ConvLSTM. The input channel of the CNN is 4, and the hidden channel is 256 with a 3×3 kernel. The hidden channel of the ConvLSTM is 128 with kernel 5×5. The output from the ConvLSTM is then fed into two linear layers activated by ReLU as the adapted task statistics. The inference network is composed of two CNNs following two linear layers. The hidden channel of the CNNs are 64 and 128, respectively, and the kernel sizes are 3×3 and 5×5, respectively. The output is then fed into a linear layer activated by ReLU and another linear layer activated by Sigmoid.
- **SNAIL** contains two layers of CNN following three attention blocks sandwiched by two TC blocks. The output is then fed into a linear layer for prediction. In detail, both CNN layers have 64 hidden channels with kernel sizes as 3×3 and 5×5, respectively. The number of filters for the TC blocks are all 128. The key sizes of the three attention blocks are 64, 256, 512, respectively, and the value sizes being 32, 128, 256, respectively.
- **cST-ML** includes a task inference network and a posterior prediction network. The task inference network is implemented with a CNN layer following a ConvLSTM layer. The input channel of the CNN is 4, and the hidden channel is 256 with a 3×3 kernel. The hidden channel of the ConvLSTM is 128 with kernel 5×5. The output from the ConvLSTM is then fed into one linear layer activated by ReLU, then the output is put into two different linear layers whose outputs work as the mean and log-variance of a Gaussian distribution respectively. The inference network shares the same structure with that in the LSTM-ML.
- **DAC-ML** has an inference network and a posterior prediction network. The inference network is composed of a domain inference network and a task inference network. To capture monthly, weekly, and daily domain statistics, the domain inference network comprises three ConvLSTM-based modules. These three modules are the same in structure as the task inference network. Specifically, each ConvLSTM-based module is implemented with a layer of CNN following a layer of ConvLSTM. The input channel of the CNN is 4, and the hidden channel is 64 with a 3×3 kernel. The hidden channel of the ConvLSTM is 128 with kernel 5×5. The three intermediate outputs from the monthly, weekly, and daily ConvLSTM networks are added together and input into a linear layer, the output is then fed into two different linear layers, respectively, to get the mean and log-variance of the domain statistics distribution. The intermediate output from the task inference ConvLSTM network is fed into one linear layer first and then into two different linear layers, respectively, to get the mean and log-variance for the task statistics prediction. The posterior prediction net shares the same structure as that in the LSTM-ML.