Transfer Knowledge Between Cities by Incremental Few-Shot Learning

Jiahao Wang[1,2], Wenxiong Li[1], Xiuxiu Qi[1(✉)], and Yuheng Ren[3]

[1] School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, China
[2] Yangtze River Delta Research Institute, University of Electronic Science and Technology of China, Huzhou, China
[3] Xiamen Construction and Climb Corporation Limited, Xiamen, China

Abstract. The objective of cross-city transfer learning methods focuses on how to effectively transfer knowledge from data-rich cities to help data-scarce cities, and solve the problem that city development levels are quite unbalanced. However, transfer-learning and meta-learning-based spatial-temporal approaches can quickly learn and adapt to (novel-) source cities, but the prior experience in base-source cities will be largely forgotten, i.e., the models may lead to catastrophic forgetting problem on base attributes. In this paper, we proposed an incremental few-shot learning based spatial-temporal model (IFS-STP), which utilized an incremental few-shot learner strives to build a generalized model that can not only transfer learned knowledge from source cities to improve the performance of spatial-temporal prediction in a target city with limited data but also prevent the catastrophic forgetting problem of source cities. We evaluate IFS-STP on traffic prediction tasks and the experience results show that our approach significantly outperforms competitive baseline models.

Keywords: Spatial-temporal prediction · Incremental few-shot learning · Meta-learning · Traffic prediction

1 Introduction

Recently, the construction of smart cities has been promoted with the application of deep learning [1, 2]. In smart urban computing, the spatial-temporal prediction problem is a fundamental problem, e.g., traffic flow, water quality, and air quality, which is the key technique in building a smart city. Traffic prediction system has drawn increasing attention due to its impacts in real-world applications such as deploy transportation resources and control traffic signal intelligently.

Previous works achieve impressive breakthroughs in the spatial-temporal prediction problem. To get around spatial-temporal prediction, basic time series models [3, 4], regression models with spatial-temporal regularizations [5], and external context features [6] are used for spatial-temporal prediction. Moreover, deep neural network models significantly improve the performance of spatial-temporal prediction [6–8] by characterizing nonlinear spatial-temporal correlations more exactly.

H. Gao and X. Wang (Eds.): CollaborateCom 2021, LNICST 407, pp. 241–257, 2021.
https://doi.org/10.1007/978-3-030-92638-0_15

However, the superior performance of these methods is conditioned on large-scale training data which are probably inaccessible in real-world applications. Data collection with unbalanced spatial distributions is the most common way to spatial-temporal prediction problems. As an example, some regions may have abundant traffic data whereas some regions may exist only a few days of traffic data for prediction.

The similarity of distributions between different cities verifies the spatial functionality is globally shared [9]. Transfer learning [10] has been studied as an effective solution to address the data insufficiency problem, by leveraging knowledge from those cities with abundant data. While previous works achieve impressive breakthroughs in the spatial-temporal prediction problem, there are at least three challenges in the existing approaches: (i) The earlier methods transferring the knowledge from only a single source city, would cause unstable results and the risk of negative transfer. (ii) Existing approaches are difficult to learn effectively with limited data due to missing values or the effects of special events (e.g., holidays). (iii) [9] using meta-learning methods with a rapid adaptation according to new information, e.g. MAML [11], but old knowledge is forgotten equally quickly. This greatly reduces the generalization ability of the model. To tackle the aforementioned practice problems, we proposed an incremental few-shot learning algorithm that enables the backbone network to transfer knowledge from multiple cities. Build on the insights of incremental learning, our motivation is to pursue novel knowledge from new-source cities and merge it with prior knowledge learned from previous experience (base-source cities) to prevent catastrophic forgetting [12]. Different from previous studies [9, 13], we aim to utilize incremental few-shot learner strives to build a generalized model that can not only transfer learned knowledge from source cities to improve the performance of spatial-temporal prediction in a target city with limited data, but also prevent the catastrophic forgetting problem of source cities. We summarize our contributions as follows.

- To the best of our knowledge, we are the first to utilize incremental few-shot learning algorithm to solve the traffic flow prediction problem.
- We propose a novel IFS-STP framework to solve the problem by combining a spatial-temporal network with the meta-learning paradigm. Moreover, we optimize a memory regularizer, describing and storing long-term spatial-temporal patterns, that reduces catastrophic forgetting from the incremental few-shot learning.
- We empirically demonstrate that IFS-STP compares favorably to state-of-art conventional methods.

The remainder of this paper is organized as follows: Sect. 2 reviews the related work. Section 3 defines some notations and formulate the problem. Section 4 introduces the details of our proposed framework of IFS-STP. Then we apply our model on five real-world datasets and conduct extensive experiments in Sect. 5. Finally, we conclude our paper in Sect. 6.

2 Related Work

In this section, we present recent researches related to our work, which mainly contains some representative works for spatial-temporal prediction and knowledge transferring. All these observations indeed motivate the work of this paper.

2.1 Spatial-Temporal Prediction

The earliest studies on spatial-temporal prediction problems almost only utilizes the basic time series data (e.g., ARIMA). Recent work further utilize some context data to make model more effective [14–16]. Wang et al. [14] propose to capture city dynamics via venue information of POI data. Wu et al. [15] study shows that external datasets (e.g., geotagged tweets, holiday and weather conditions) can be helpful. Tong et al. [16] adopt a simple linear model with very high-dimensional features in predicting the Unit Original Taxi Demand.

Recently, various deep learning methods have attracted much attention from many researchers and have been applied to deal with the problem of spatial-temporal prediction. Yi et al. [6] proposed a DNN-based distributed fusion network to fuse heterogeneous urban data. Yu et al. [17] build a deep neural network based on LSTM units to forecast urban traffic. Yao et al. [18] propose a novel Spatial-Temporal Dynamic Network to model both spatial and temporal information with CNN and RNN.

2.2 Transfer Learning

Different from the traditional spatial-temporal prediction tasks which all rely on large-scale training data, we aim to utilize meta-learning paradigm to transfer learned knowledge from source cities to a target city with limited data samples to improve the performance of spatial-temporal prediction of the target city.

Recently, few-shot learning methods have caught the attention of researchers. These methods aim to transfer shared knowledge from multiple training tasks to a new task for quick adaptation. [19] proposed a novel few-shot learning method to especially adopt a deep neural network—meta-transfer learning. Jiang et al. [20] proposed to extract multi-scale features and learns the relations between samples to achieve classification. In addition, this study also proposed a new loss function to optimize the model. But a key challenge for meta-learning methods is catastrophic forgetting [12], i.e., the model forgets the learned knowledge. Incremental Few-Shot (IFS) learning is also known as few-shot learning. Different from previous meta-learning methods, IFS will add basic tasks to the query set during the evaluation stage, and the classifier will be augmented to include all tasks. Ren et al. [21] propose to add a learned regularizer to help identify new classes while remembering old classes without having to review the original training data.

However, only a few attempts have been made on transferring knowledge of space. Wei et al. [22] propose to construct a regularization that can transfer the knowledge between cities. For predicting traffic flow, Yao et al. [9] propose a framework to solve spatial-temporal prediction by constructing a memory regularizer, which can learn a global memory from all source cities. But this method is still based on MAML.

3 Preliminaries

In this section, we first briefly and formally define the spatial-temporal prediction problem. We following previous works [9, 13], and propose cross-city transfer learning methods for spatial-temporal prediction, which solve the problem of the city development levels are quite unbalanced. The objective of cross-city transfer learning methods is to predict a certain type of service data (e.g., crowd flow) in a data-scarce city (target cities) by transferring knowledge learned from a data-rich city (source cities).

Definition 1. Region. For consistency to prior works [7–9, 23], we partition a city c into the $W_c \times H_c$ size (e.g., 1 km × 1 km) grid map with M regions in total ($M = W_c \times H_c$). We take each grid as a region r and r_{ij} denotes a city region with coordinates of (i,j). The whole set of regions in the city c is denoted as $\mathbb{G}_c$.

Definition 2. Time Series. In a city c, we split the time period (e.g., 1 year) into equal-length continuous time intervals. To be more specific, we denote the current/last time-stamp as d_c and the set of data time-stamps of c can be defined as:

$$\mathbb{D}_c = [d_c - |\mathcal{D}_c| + 1, d_c] \tag{1}$$

Where $|\mathcal{D}_c|$ is the number of time-stamps and $\mathbb{D}_c$ is consist of $|\mathcal{D}_c|$ equal-length time intervals (e.g., 1 h). At a specific time-stamp d_c, we define the spatial-temporal series in city c as follows:

$$\mathcal{Y}_c = \{y_{r_c,d_c} | r_c \in \mathbb{G}_c, d_c \in \mathbb{D}_c\} \tag{2}$$

Where y_{r_c,d_c} is the spatial-temporal information. It's a most common way to model a variety of urban data in reality, e.g., traffic demand and air quality.

Definition 3. Spatial-Temporal Data. Given a set of source cities $\mathcal{C}_\mathcal{S} = \{c_1, c_2, \ldots, c_o\}$ with rich data and a target city $c_k$with limited data, i.e., $|\mathbb{D}_{\mathcal{C}_\mathcal{S}}| \gg |\mathbb{D}_{\mathcal{C}_\mathcal{K}}|$. In this aforementioned case, we formalize the problem.

Definition 4. Problem Definition. Our goal is to learn a spatial-temporal network $\mathcal{F}_\theta(\cdot)$ with parameters θ to predict the spatial-temporal data in target city $c_k$at the next time-stamp $d_{c_k} + 1$. Thereby, the problem can be calculated as predicting the spatial-temporal information that maximizes the conditional probability:

$$\tilde{y}_{r_{c_k},d_{c_k}+1} = \arg \max_{y_{r_{c_k},d_{c_k}+1}} p(y_{r_{c_k},d_{c_k}+1} | \mathcal{Y}_{c_k}, \mathcal{F}_\theta) \tag{3}$$

Then, we can formulate the error function of base-learner for each city c as:

$$\begin{aligned} &\min_{\mathcal{F}_\theta} \ error(\tilde{y}_{r_{c_k},d_{c_k}+1}, y_{r_{c_k},d_{c_k}+1}), \\ &where\ \tilde{y}_{r_{c_k},d_{c_k}+1} = \mathcal{F}_\theta(c_k, c_s), \quad |\mathbb{D}_{c_s}| \gg |\mathbb{D}_{c_k}|, \end{aligned} \tag{4}$$

According to the real application requirement, the error metric can be mean absolute error, root mean squared error (RMSE), etc.

Definition 5. Application. In this paper, we use traffic prediction tasks to illustrate the aforementioned problem concretely. Similar as the previous traffic volume prediction studies in [8, 9, 23], each individual trip as the important part of the whole city traffic is always departs from a region, and then arrives at the destination region after a period of time. The start/end traffic volume in a region as the number of trips departing/arriving from/in the region at a fixed time interval. Therefore, our work is aim to predict the start and end volume of taxi at time interval $d_{c_k} + 1$.

4 Methodology

4.1 The Spatial-Temporal Network Architecture (ST-Net)

Recent researches [7, 9, 13, 23, 24] utilize convolution neural networks (CNN) and long short-term memory (LSTM) as the basic components of the neural network to learn spatial-temporal patterns and have achieved superior results. Thus, we follow previous works [7, 9] that use CNN to capture the spatial interactions between regions and an LSTM to learn sequential dependency, as shown in Fig. 1.

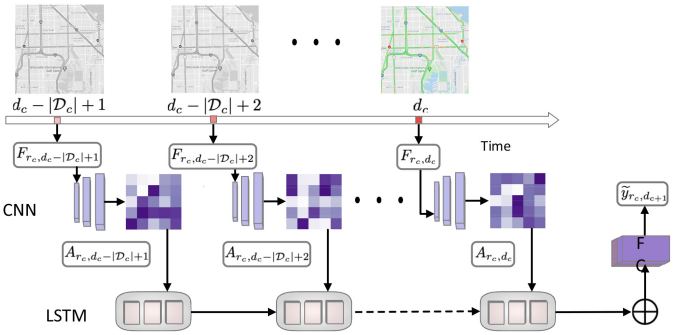

Fig. 1. The architecture of the backbone network.

Convolution: Intuitively, the traffic volume in nearby regions may affect each other, and the CNN can be effectively handled this situation. Specifically, in city c at each time interval d_c, we treat the spatial-temporal value of region r_c and its surrounding neighbors as a $S \times S$ image with v channels $F_{r_c,d_c} \in \mathbb{R}^{S \times S \times v}$. In our work, we set $v = 2$, i.e., we jointly predict taxi volume, one channel contains the start volume information, another one contains end volume information. The CNN takes F_{r_c,d_c} as input $F^0_{r_c,d_c}$, and feeds it into L convolutional layers. The formulation of each convolutional layer l is defined as follows:

$$F^{l}_{r_c,d_c} = \sigma\left(\mathbf{W}^{l}_{r} * F^{l-1}_{r_c,d_c} + \mathbf{b}^{l}_{r}\right), \tag{5}$$

$\sigma = RELU(\cdot)$ is an activation function and $*$ denotes the convolutional operation. $\mathbf{W}^{l}_{r}$ and $\mathbf{b}^{l}_{r}$ are two learnable parameters in the l-th convolution layer. After stacking L convolutional layers, a fully connected layer following a flatten layer is used to infer the spatial representation A_{r_c,d_c} for region r_c.

Temporal: In the sequel, we use LSTM to capture the temporal sequential dependency and model sequential relations in the time series. The memory cell $\boldsymbol{c}_d$ at time interval d in LSTM is an accumulator of the state information. To be more specific, in each time interval d_c, we take A_{r_c,d_c} as LSTM input, Ultimately, $\boldsymbol{h}_{r_c,d_c}$ from LSTM represents the spatial and temporal representations of region r_c. At last $\tilde{y}_{r_c,d_c+1}$ can be accessed by:

$$\tilde{y}_{r_c,d_c+1} = \tanh\left(\boldsymbol{W}_a \boldsymbol{h}_{r_c,d_c} + \mathbf{b}_a\right) \tag{6}$$

Where $\mathbf{W}_a$ and $\mathbf{b}_a$ are learnable parameters. The output of our backbone framework is scaled to $(-1, 1)$ via a $\tanh(\cdot)$ function since we normalize the value of start and end volume. We later demoralize the prediction to get the actual demand values. Moreover, the loss function for each city c we used is defined as:

$$\mathcal{L}_c = \sum\nolimits_{r_c} \sum\nolimits_{k_c} \left(\mathcal{F}_\theta\left(f_{r_c,d_c}\right)\right) - y_{r_c,d_c+1})^2 \tag{7}$$

where θ is the learnable parameters in the spatial-temporal network.

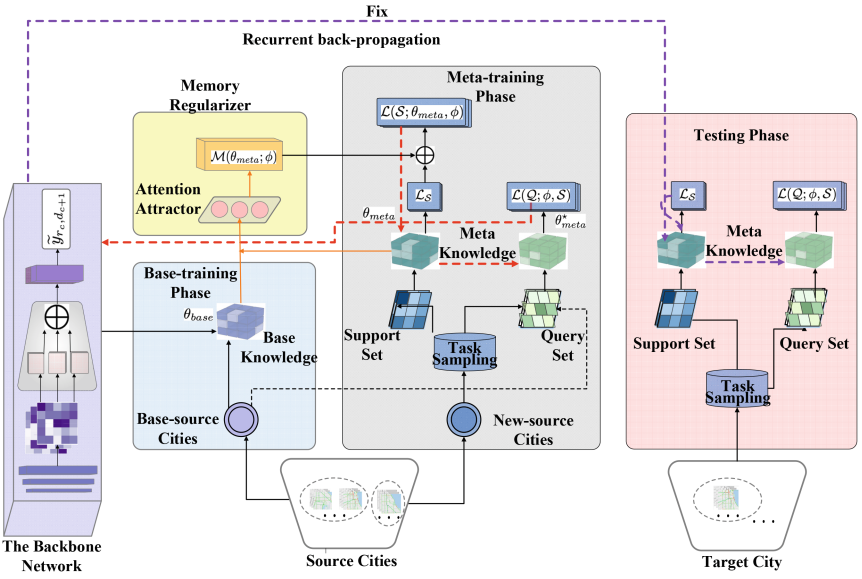

Fig. 2. The architecture and parameter update process (i.e., knowledge transfer process) of our proposed model

4.2 Incremental Few-Shot Learning

In this subsection, we proposed an incremental few-shot learning algorithm that enables ST-Net to transfer knowledge from multiple cities. Albeit used meta-learning methods with a rapid adaptation according to new information, such as MAML [11], Yao used a meta-learning approach for spatial-temporal prediction [9], but old knowledge is forgotten equally quickly. The main issue of incremental learning [21, 25, 26] is to overcome catastrophic forgetting [12], after learning the new tasks, the information of the old task is missed, resulting in the poor performance of the model. Build on the insights of incremental learning, our motivation is to pursue novel knowledge from new-source cities and merge it with prior knowledge learned from previous experience (base-source cities) to prevent catastrophic forgetting of the base-source cities. Thus, we detail our meta-learning approach to incremental few-shot learning in three parts: (i) Base-training phase, pre-training a backbone network (ST-Net) on a set of basic cities; (ii) Incremental Few-Shot task, built upon the pre-trained model, meta-training is done with episodes(tasks) form of training with our meta-learnd memory loader; (iii) Meta-training phase, leverage memory regularizer to extract the mixture of the base and novel knowledge. For better illustration, the incremental few-shot attribute learning process of the proposed model is shown in Fig. 2. Details of these stages follow.

Base-Training Phase: $\mathcal{C}_{\mathcal{S}}$ denotes the set of source cities $\mathcal{C}_{\mathcal{S}} = \{c_1, c_2, \ldots, c_o\}$. It is split into base-source cities $\mathcal{C}_{\mathcal{S}}^a$ and new-source cities $\mathcal{C}_{\mathcal{S}}^b$ for base-training and meta training, i.e., $\mathcal{C}_{\mathcal{S}} = \mathcal{C}_{\mathcal{S}}^a \cup \mathcal{C}_{\mathcal{S}}^b$. We learn the backbone network (ST-Net) $\mathcal{F}_\theta(\cdot)$ on $\mathcal{C}_{\mathcal{S}}^a$. The purpose of this stage is to learn both a good base learner and a good representation. The parameters of the backbone network θ_{base} are learned in this stage and will be fixed after base-training.

Incremental Few-Shot Task: We define the task training strategy widely used by existing meta-learning based few-shot learning models [9, 11, 27]. Meta-learning leverage the shared structure across different tasks from the task distribution $p(\mathcal{T})$, learning prior knowledge for new tasks. Each task $\mathcal{T}_i$ splits the sampled data-points into support set $\mathcal{S}$ used for training the model with a memory loader and the query sets $\mathcal{Q}$ for measuring whether or not training was effective.

Concretely, under the few-shot setting, we give a sample set from a set of new-source cities $\mathcal{C}_{\mathcal{S}}^b$. For each $\left|\mathcal{C}_{\mathcal{S}}^b\right|$-way η-shot task, there are $\left|\mathcal{C}_{\mathcal{S}}^b\right|$ novel source cities disjoint from the base-source cities. Each new-source city has η and ν datapoints to form the support set $\mathcal{S}^b$ and novel-query set $\mathcal{Q}^b$ respectively. Therefore, we have subtask $(\mathcal{S}^b, \mathcal{Q}^b)$, in which we learn on $\mathcal{S}^b$ whose learnable parameters θ_{meta} are called the fast weights as they are only used during this task. The query set $\mathcal{Q}$ consists of samples not only from the novel but also base source cities, i.e., a $(\left|\mathcal{C}_{\mathcal{S}}^a\right| + \left|\mathcal{C}_{\mathcal{S}}^b\right|)$-way. Formally, we sampled ν datapoints from $\mathcal{C}_{\mathcal{S}}^a$ to form the base-query set $\mathcal{Q}^a$ and added to $\mathcal{Q}^b$ to form the query set $\mathcal{Q} = \mathcal{Q}^a \cup \mathcal{Q}^b$. For each task, learnable parameters are updated by computing loss during training on $\mathcal{Q}_i$. As indicated aforementioned, we use the following steps to iteratively sample tasks $\mathcal{T} = \mathcal{S}^b \cup \mathcal{Q}$ and use them to train the model:

- For each new-source city $c \in \mathcal{C}_{\mathcal{S}}^{b}$, randomly sample η datapoints to form a support set $\mathcal{S}_i^b$;
- υ data-points ($\notin \mathcal{S}_i^b$) were extracted from base-source cities $\mathcal{C}_{\mathcal{S}}^{a}$ and new-source cities $\mathcal{C}_{\mathcal{S}}^{b}$ respectively to form $\mathcal{Q}_i^a$ and $\mathcal{Q}_i^b$, where $\mathcal{Q}_i = \mathcal{Q}_i^a \cup \mathcal{Q}_i^b$.
- Repeat step (1) and (2) for $|\mathcal{T}| = \mathcal{I}$ times.

Meta-training Phase: In the meta-training phase, we learn the meta-parameters in order to minimize the joint prediction loss on $\mathcal{Q}_i = \mathcal{Q}_i^a \cup \mathcal{Q}_i^b$. Here, we design a memory regularizer $\mathcal{M}(\cdot, \phi)$ such that the fast weights are learned via minimizing the loss $\mathcal{L}(\theta_{meta}, \mathcal{S}) + \mathcal{M}(\theta_{meta}, \phi)$ where $\mathcal{L}(\theta_{meta}, \mathcal{S})$ is root mean square error for traffic volume prediction. For the meta-learning stage, the meta-parameters of meta-learner ϕ are optimized by iterative updating on $\mathcal{S}(\theta_{meta} \rightarrow \theta_{meta}^{*})$, which are exactly the knowledge that encrypts spatial-temporal correlations. In our model, meta-parameters are encapsulated via the attention-attractor network, which produces regularizers for the fast weights in the few-shot learning objective.

4.3 Memory Regularizer Combine Base and Novel Knowledge

The similarity of distributions between different cities verifies the spatial functionality is globally shared [9]. However, a target city still suffers from certain constraints.

To this end, we proposed a spatial-temporal memory regularizer to load the mixture weight vectors θ_{base} and θ_{meta}.We start with an attention mechanism to encode base information (knowledge) θ_{base}. Here we taken the learned vector ($\mathcal{U}_w = \theta_{base,\omega}$) of each base class stored in the memory matrix $\mathcal{U} = \{\mathcal{U}_1, \mathcal{U}_2, \ldots, \mathcal{U}_{|\mathcal{C}_{\mathcal{S}}^{a}|}\}$. The attention mechanism assigns the attention weight $\mathcal{V}_{\omega',\omega}$ to base weight vectors as follows ($\omega \in \mathcal{C}_{\mathcal{S}}^{a}$):

$$\begin{aligned} \alpha_{\omega} &= \tau \mathcal{A}\left(\frac{1}{\eta}\sum\nolimits_{j \in \mathcal{S}_i} \mathbf{h}_j, \theta_{base,\omega}\right), \\ \mathcal{V}_{\omega',\omega} &= \frac{\exp(\alpha_{\omega})}{\sum_{\omega' \in \mathcal{C}_{\mathcal{S}}^{b}} \exp(\alpha_{\omega'})}, \\ \mathcal{U}_{\omega'} &= \sum\nolimits_{\omega \in \mathcal{C}_{\mathcal{S}}^{a}} \mathcal{V}_{\omega',\omega}\theta_{base,\omega} + U_0, \end{aligned} \tag{8}$$

where $\mathcal{A}$ is the cosine similarity function, $\mathbf{h}_j$ represents the spatial-temporal representations of inputs in the support set $\mathcal{S}$ and τ is a learnable temperature scalar. Thus, the attention vector is used to compute the memory matrix, which stored a learned

attractor vector of each base-source city, where $\mathcal{U}_0$ is an embedding vector and serves as a bias for the attractor. Thus we can derive the memory regularizer. The key feature of memory regularizer is the regularization term $\mathcal{M}(\theta_{meta}, \phi)$:

$$\mathcal{M} = \sum\nolimits_{\omega' \in \left|\mathcal{C}_{\mathcal{S}}^{b}\right|} \left(\theta_{meta,\omega'} - u_{\omega'}\right)^{\mathrm{T}} \mathrm{diag}(\exp(\gamma)) \left(\theta_{meta,\omega'} - u_{\omega'}\right) \tag{9}$$

where $u_{\omega'}$ is the so-called attractor and $\theta_{meta,\omega'}$ is the $\omega^{'}$-th column of θ_{meta}. This sum of squared Mahalanobis distances from the attractors adds a bias to the learning signal arriving solely from novel-source cities. To conclude, the meta-parameters ϕ include $\mathcal{U}_0$, γ, and τ, which have the same dimension as base parameters θ_{base}. In meta-training phase, we can formulate the task objective as:

$$\mathcal{L}(\mathcal{S}_i; \theta_{meta}, \phi) = -\frac{1}{\left|\mathcal{C}_{\mathcal{S}}^{b}\right|\eta} \sum\nolimits_{j \in \mathcal{S}_i} y_{i,j}^{\mathcal{S}} \log \tilde{y}_{i,j}^{\mathcal{S}} + \mathcal{M}(\theta_{meta}, \phi) \tag{10}$$

Once the support set is processed, θ_{meta}^{*} is the optimal parameters that minimize the regularized objective in Eq. 10. During testing on the joint prediction of both base and novel source cities in query set $\mathcal{Q}$, $\tilde{y}_v^{\mathcal{S}} = \mathcal{F}\left(F_v; \theta_{meta}^{*}\right)$.

During meta-learning, for each task, ϕ are updated to minimize an expected loss of the query set $\mathcal{Q}$ which contains both base and novel classes:

$$\mathcal{L}(\mathcal{Q}_i; \phi, \mathcal{S}_i) = \mathrm{argmin} - \log p\left(\mathcal{Q}_i \middle| \theta_{meta}^{*}(\phi, \mathcal{S}_i)\right) \tag{11}$$

The pseudo-codes for meta-training phase is outlined in Algorithm 1.

4.4 Parameter Optimization (Knowledge Transfer)

In each task, we need to minimize $\mathcal{L}(\mathcal{S}_i)$ to obtain θ_{meta}^{*} through an iterative optimizer. More importantly, need to transfer all learned knowledge to model and optimize it to improve the prediction of the target cities. For this reason, the question is how to efficiently compute $\frac{\partial \theta_{meta}^{*}}{\partial \phi}$, i.e., back-propagating through the optimization. In our work, we use the recurrent back-propagation (RBP) algorithm [28, 29] to efficiently back-propagate through the fixed point. The fast weight updating via a fixed number of gradient descent can be formulate as:

$$\theta_{meta}^{\epsilon+1} = \theta_{meta}^{\epsilon} - \beta \nabla \mathcal{L}\left(\mathcal{S}_i, \theta_{meta}^{\epsilon}\right) \tag{12}$$

Algorithm 1: IFS-STP Training Algorithm.

Input: The set of source cities $\mathcal{C}_S = \mathcal{C}_S^a \cup \mathcal{C}_S^b$, the set of target cities $\mathcal{C}_k$.
Output: Traffic volume prediction of each target city.
/* Base-Training Phase. */
Learn the parameters of backbone network θ_{base} on $\mathcal{C}_S^a$
/* Meta-Training Phase. */
for $i = 1, \cdots, \mathcal{I}$ **do**
  $\{\mathcal{S}_i^b, \mathcal{Q}_i^b\} \leftarrow$Sample support/novel-query set ($\mathcal{C}_S^b$);
  $\mathcal{Q}_i^{a+b} \leftarrow$ Sample base-query set ($\mathcal{C}_S^a$) $\cup$ $\mathcal{Q}_i^b$
  Get the task $\mathcal{T}_i = \{\mathcal{S}_i^b, \mathcal{Q}_i\}$
  while θ_{meta} not converged **do**
    Compute memory knowledge $\mathcal{M}$ by Eq 8 and 9 using θ_{base} and θ_{meta};
    Compute $\mathcal{L}(\mathcal{S}_i; \theta_{meta}, \phi)$by Eq 10 using $\mathcal{S}_i$ and $\mathcal{M}(\theta_{meta}, \phi)$;
    Update θ_{meta} through optimization step: $\theta_{meta} \leftarrow \nabla_{\theta_{meta}} \mathcal{L}(\mathcal{S}_i)$
  end
  Evaluate task loss $\mathcal{L}(\mathcal{Q}_i; \phi; \mathcal{S}_i)$ on query set $\mathcal{Q}_i$ by Eq 11;
  /* Backprop through the above optimization via RBP */
  /* A dummy gradient descent step */
  $\theta'_{meta} \leftarrow \theta_{meta} - \beta \nabla_{\theta_{meta}} \mathcal{L}(\mathcal{S}_i)$
  $\mathcal{J} \leftarrow \frac{\partial \theta'_{meta}}{\partial \theta_{meta}}$; $\rho \leftarrow \frac{\partial \mathcal{L}(\mathcal{Q}_i)}{\partial \theta_{meta}}$;
  Initialization $g \leftarrow \rho$;
  while g not converged **do**
    $\tilde{\rho} = \mathcal{J}^T \rho - \delta \rho$
    $g \leftarrow g + \tilde{\rho}$;
  end
  Update the network parameters: $\phi^* \leftarrow g^T \frac{\partial \theta'_{meta}}{\partial \phi}$
end
  /* Evaluating IFS-STP on target cities. */
for each target city $c_k \in \mathcal{C}_{\mathcal{K}}$ **do**
  $\mathcal{S} \leftarrow$ Sample a support set (c_k);
  Compute $\mathcal{L}(\mathcal{S}; \theta_{meta}, \phi^*)$ by Eq 12 and adapt parameters θ_{meta} through optimization step: $\theta_{meta} \leftarrow \nabla_{\theta_{meta}} \mathcal{L}(\mathcal{S})$
  Sample new series to form a query set $\mathcal{Q}$ and predict.
end

where β is the step size. We update θ_{meta} by a vanilla gradient descent process with step size β. The difference between two steps ψ can be written as:

$$\psi(\theta_{meta}^{\epsilon}) = (\theta_{meta}^{\epsilon}) - \mathcal{Z}(\theta_{meta}^{\epsilon}) \tag{13}$$

where $\mathcal{Z}(\theta_{meta}^{\epsilon}) = \theta_{meta}^{\epsilon+1}$ is the update function parameterized by θ_{meta}. At the fixed point, $\psi(\theta_{meta}^*) = 0$, using the total derivative and the dependence of θ_{meta}^* on ϕ we can get:

$$\frac{\partial \psi(\theta^*_{meta})}{\partial \phi} = \left(\mathcal{I} - \mathcal{J}^T_{\mathcal{Z},\theta^*_{meta}}\right)^{-1} \frac{\partial \theta^*_{meta}}{\partial \phi} - \frac{\partial \mathcal{Z}}{\partial \phi} = 0 \tag{14}$$

where $\mathcal{J}_{\mathcal{Z},\theta^*_{meta}}$ is the Jacobian matrix of $\mathcal{Z}$wvaluated at θ^*_{meta}. Since $\mathcal{I} - \mathcal{J}^T_{\mathcal{Z},\theta^*_{meta}}$ is invertible, we reformulate Eq. 14 as:

$$\frac{\partial \theta^*_{meta}}{\partial \phi} = \left(\mathcal{I} - \mathcal{J}^T_{\mathcal{Z},\theta^*_{meta}}\right)^{-1} \frac{\partial \mathcal{Z}}{\partial \phi} \tag{15}$$

It is worth noting that, Eq. 13, Eq. 14, and Eq. 15 apply the Implicit Function Theorem, which guarantees the existence and uniqueness of an implicit function. Similar to previous work [9, 28], we use the Neumann series $(\mathcal{I} - \mathcal{J}^T)^{-1}\rho = \sum_{n=0}^{\infty}(\mathcal{J}^T)^n \rho \equiv$to compute the matrix-inverse vector product $(\mathcal{I} - \mathcal{J}^T)^{-1}\rho$. To avert numerical instability caused by directly applying the Neumann RBP algorithm, it proposed to add a damping term δ to $(\mathcal{I} - \mathcal{J}^T)$. The results is update by: $\tilde{\rho}^{(n)} = (\mathcal{J}^T - \delta\mathcal{I})^n \rho$, where $\delta = 0.1$.

5 Experiment

In this Section, we use traffic volume prediction (i.e., taxi volume prediction), an important type of spatial-temporal prediction task in urban computing. Following previous studies on traffic prediction [8, 9, 23], we aim to predict the start and end volume of taxi at each time interval for each region to tackle the following issues:

- **Q1**: How does IFS-STP perform compared with the traditional traffic volume prediction models, transfer-learning based models and meta-learning based approaches?
- **Q2**: How do different parts in IFS-STP affect predictive performance?
- **Q3**: How do the optimization method affect the performance of IFS-STP?

5.1 Datasets

We use the datasets and experimental setup in [9]. We conduct experiments for the domain application of taxi volume prediction, and collect datasets from five different cities, i.e., New York City (NYC), Washington (D.C.), Chicago (CHI), Porto and Boston (BOS).

Table 1. Descriptive statistics of all datasets.

Task	City	Trip records	Map size	Time span
Taxi	NYC	6,748,857	10 × 20	1/1/15–7/1/15
	D.C.	8,151,077	16 × 16	5/1/15–1/1/16
	Porto	1,710,671	20 × 10	7/1/13–6/30/14
	CHI	124,820	15 × 18	9/1/13–11/1/14
	BOS	839,897	18 × 15	10/1/12–10/31/12

Taxi Volume Prediction: For the task, we evaluate our proposed method on five real-world mobility datasets, which collecting from five different cities, i.e., NYC, D.C., CHI, Porto, and BOS. Remarkably, we use $\mathcal{C}_{\mathcal{S}} = \{\text{NYC}, \text{D.C.}, \text{Porto}\}$ as the source cities (where $\mathcal{C}_{\mathcal{S}}^{a} = \{\text{NYC}\}, \mathcal{C}_{\mathcal{S}}^{b} = \{\text{D.C.}, \text{Porto}\}$) and $\mathcal{C}_{\mathcal{K}} = \{\text{CHI}, \text{BOS}\}$ (we only predict start volume in BOS) as target cities.

5.2 Data Preprocessing

Table 1 details the statistics of all datasets. We spatially partition a city into a grid map (regions). For example, in taxi volume prediction, the grid map size of NYC, D.C., CHI, BOS, Porto are 10×20, 16×16, 15×18, 18×15, 20×10, respectively. In addition, for each source city, we select 80% data for training/validation, and the rest for testing. For each target city, we select the 1-day, 3-day and 7-day data for training, and the rest for testing. And the time intervals of the traffic prediction task is 1 h.

5.3 Baselines

We compare our model IFS-STP with the following three group benchmark baselines.

Traditional traffic prediction methods:

- **Historical average (HA)**: Historical average is the traditional time-series prediction method, which predicts spatial-temporal value based on the average value of the previous relative time interval.
- **Autoregressive integrated moving average (ARIMA)** [30]: ARIMA is a widely used time-series prediction method in statistics.
- **ST-Net***: ST-Net is a deep learning based method for traffic prediction.

Transfer-learning based traffic models:

- **Fine-tuning Methods (FT)**: FT method is a transfer baseline. Following [13], we use single-source fine-tune (Single-FT) and multi-source fine-tune (Multi-FT) methods.
- **RegionTrans** [31]: RegionTrans proposed a novel deep spatial-temporal transfer learning framework for traffic flow prediction.

Meta-learning based traffic approaches:

- **MAML** [11]: MAML proposed a meta-learning method based on learning easily adaptable model parameters through gradient descent.
- **MetaST** [9]: MetaST leverages learned knowledge from multiple source cities to help the prediction in target data-scarce cities for spatial-temporal prediction.

5.4 Evaluation Metric

We evaluate our model using the widely used metrics: Root Mean Squared Error (RMSE), which is defined as follows:

$$\mathrm{RMSE} = \frac{1}{|N|}\sqrt{\sum_{r_{c_k}}\sum_{d_{c_k}}\left(\tilde{y}_{r_{c_k},d_{c_k}+1} - y_{r_{c_k},d_{c_k}+1}\right)^2} \tag{16}$$

where $\tilde{y}_{r_{c_k},d_{c_k}+1}$and $y_{r_{c_k},d_{c_k}+1}$mean the prediction value and ground truth of region at time interval $d+1$, respectively. $|N|$ is the total number of samples in target city c_k.

5.5 Experimental Settings

In the backbone network, we set all convolution kernel size to 3×3 with 64 filters, the size of each neighborhood as 7×7 for convolution component and IFS-STP does not use other external features; we set the number of steps in LSTM as 8, and the dimension of hidden representation of LSTM as 128 in temporal component. In meta-training phase of taxi volume prediction, we use Adam optimizer with a learning rate of 1e–5. Moreover, in order to alleviate overfitting, we use early-stop in our experiments.

Table 2. Evaluation result compared to baseline.

Models		Chicago(CHI)						Boston(BOS)		
		Start			End			Start		
		1-day	3-day	7-day	1-day	3-day	7-day	1-day	3-day	7-day
HA		2.83	2.36	2.18	2.67	2.28	2.13	11.07	9.13	7.71
ARIMA		3.19	2.76	2.71	2.93	2.43	2.41	2.93	2.43	2.41
ST-Net		10.51	6.04	3.89	10.51	6.04	3.89	10.51	6.04	3.89
Single-FT	NYC CHI	2.72	2.06	1.76	2.57	1.87	1.60	12.86	9.50	8.11
	D.C. CHI	3.90	2.62	2.05	4.17	2.19	2.15	15.88	10.07	10.16
	Porto CHI	2.57	1.87	1.60	2.87	2.03	1.74	12.91	8.54	8.08
Multi-FT		2.18	1.89	1.60	2.20	2.08	1.69	8.50	8.22	8.01
Region Trans	NYC CHI	2.53	2.01	1.69	2.83	2.56	1.72	11.98	9.46	7.95
	D.C. CHI	3.87	2.51	2.04	3.95	2.16	2.03	14.76	9.23	10.12
	Porto CHI	2.45	1.83	1.60	2.85	1.98	1.73	8.43	8.09	8.07
MAML		2.01	1.78	1.52	2.10	1.92	1.66	8.18	7.60	7.25
MetaML		1.95	1.70	1.48	2.04	1.79	1.65	7.81	6.97	6.58
IFS-STP		1.91	1.65	1.46	1.99	1.70	1.65	7.67	6.71	6.37

5.6 Experimental Result

We evaluate IFS-STP on taxi volume prediction. To ensure a fair comparison, we compare the baseline results obtained in [9] and use the experimental settings in [9]. We average the results of the 10 test times. Thus Table 2 shows the performance of our proposed method as compared to all other competing methods in the taxi datasets, respectively. According to Table 2, our proposed IFS-STP model implements the lowest RMSE on test datasets, which significantly outperforms all baselines. To validate, we evaluate our IFS-STP model on the standard traffic prediction methods and meta-learning benchmark.

The traditional traffic volume prediction methods, e.g., HA and ARIMA, are perform ineffective, because these methods only rely on the historical records of predicted value and overlook the spatial and temporal features. Nonetheless, in some situations (1-day data for training), the traditional traffic volume prediction methods still have competitive results when compared with ST-Net and some Single-FT, Region Trans models, the reason lies in the traffic data is periodicity, so that we could predict traffic start/end volume from limited data.

For transfer-learning based methods, e.g., Single-FT, Multi-FT and Region Trans models, construct the deep spatial-temporal transfer-learning framework, which can predict future traffic volume in target cities by transferring knowledge from source cities. The experiment results in Table 2 show that comparing with ST-Net the knowledge transfer between cities is effective for prediction. To be more specific, in contrast to Single-FT and RegionTrans, Multi-FT is preferable in most cases, the possible reason is that training on multiple source cities can obtain the diversity information from the source domain.

Comparing with traditional and transfer-learning based models, all meta-learning methods, e.g., MAML, MetaST, and IFS-STP achieve better performance. This is because meta-learning methods learn the initialization of multiple cities based on multiple cities, which can learn the common representation of spatial-temporal tasks in different cities and quickly adapt to new tasks. In particular, our proposed model IFS-STP achieves the best performance in all experimental settings. For every learning model, the performance improves with more training data (e.g., 3-day, 7-day). On all benchmarks, our model still shows a significant margin over the prior works, which signifies that our iterative model can solve the few-shot objective till convergence is better than baselines and that recurrent back-propagation is an effective and modular tool for learning in a general meta-learning setting.

5.7 Parameter Sensitivity

We conduct a sensitivity analysis to investigate the influence of two important parameters of IFS-STP. In this work, we report the results on Chicago dataset, but note that the results are similar on other datasets. Figure 3 illustrates the impact of the optimization method RBP on RMSE. We use the scenario of 3-day data for sensitivity analysis.

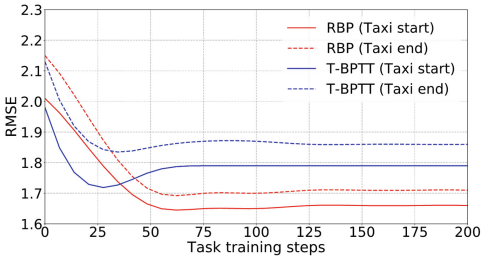

Fig. 3. Using T-BPTT and RBP to learn the proposed model.

In this study, we use the truncated BPTT [32] (T-BPTT) as the contrast experiment, which is a commonly used algorithm in many recent meta-learning approaches [11, 33]. TBPTT is optimized for gradient-based T-step optimization, when T is small, the training objective may have significant deviation. Remarkable, the RBP algorithm has the same time complexity compared to truncated BPTT given the same number of unrolled steps, but RBP does not have to store intermediate activations. From Fig. 3, the performance of TBPTT learned models are comparable to RBP. However, when solved to convergence at test time, the performance of T-BPTT models drops significantly. The reason may be they are only guaranteed to work well for a certain number of steps and failed to learn a good regularizer.

6 Conclusions

In this paper, we show that our model trained with an incremental few-shot learning curriculum achieves the top performance for tackling traffic prediction problems. We start with the spatial-temporal network architecture with learnable parameters, and then define the setup of incremental few-shot learning. In addition, we proposed a memory regularizer to store the prior knowledge about source cities and extend it to the target city, so as to improve the prediction accuracy of the target city.

For future work, we plan to investigate from three directions: (1) we plan to organize the hierarchical memory to improve the long-term spatial-temporal characteristics of the model. (2) We plan to further consider basic network structure and combine Graph Neural Network with our model. (3) We plan to perform an interpretative analysis of our model and analyze how the information is transferred.

Acknowledgement. This work is supported by UESTC-ZHIXIAOJING Joint Research Center of Smart Home (No. H04W210180), Neijiang technology incubation and transformation Funds (No. 2021KJFH004).

References

1. Liu, Y., Zhao, K., Cong, G.: Efficient similar region search with deep metric learning. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1850–1859 (2018)
2. Wei, H., Zheng, G., Yao, H., Li, Z.: Intellilight: A reinforcement learning approach for intelligent traffic light control. In: The 24th ACM SIGKDD, pp. 2496–2505 (2018)
3. Lippi, M., Bertini, M., Frasconi, P.: Short-term traffic flow forecasting: an experimental comparison of time-series analysis and supervised learning. IEEE Trans. Intell. Trans. Syst. **14**(2), 871–882 (2013)
4. Moreira-Matias, L., Gama, J., Ferreira, M., Mendes-Moreira, J., Damas, L.: Predicting taxi–passenger demand using streaming data. IEEE Trans. Intell. Trans. Syst. **14**(3), 1393–1402 (2013)
5. Zheng, J., Ni, L.M.: Time-dependent trajectory regression on road networks via multi-task learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, 27(1), (2013)
6. Yi, X., Zhang, J., Wang, Z., Li, T., Zheng, Y.: Deep distributed fusion network for air quality prediction. In: Proceedings of the 24th ACM SIGKDD, pp. 965–973 (2018)
7. Yao, H., et al.: Deep multi-view spatial-temporal network for taxi demand prediction, In: AAAI, vol. 32(1), pp. 2588–2595 (2018)
8. Zhang, J., Zheng, Y., Qi, D.: Deep spatio-temporal residual networks for citywide crowd flows prediction. In: Thirty-first AAAI Conference, vol. 31(1), pp. 1655–1661 (2017)
9. Yao, H., Liu, Y., Wei, Y., Tang, X., Li, Z.: Learning from multiple cities: a meta-learning approach for spatial-temporal prediction. In: WWW Conference, pp. 2181–2191 (2019)
10. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. IEEE Trans. Neural Networks **22**(2), 199–210 (2011)
11. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International Conference on Machine Learning, pp. 1126–1135 (2017)
12. Goodfellow, I.J., Mirza, M., Courville, A., Bengio, Y.: An empirical investigation of catastrophic forgetting in gradient-based neural networks. Stat. **6**, 1050 (2014)
13. Wang, L., Geng, X., Ma, X., Liu, F., Yang, Q.: Cross-city transfer learning for deep spatio-temporal prediction. In: Proceedings of the 28th IJCAI, pp. 1893–1899 (2019)
14. Wang, H., Yao, H., Kifer, D., Graif, C., Li, Z.: Non-Stationary model for crime rate inference using modern urban data. IEEE Trans, Big Data **5**(2), 180–194 (2017)
15. Fei, W., Wang, H., Li, Z.: Interpreting traffic dynamics using ubiquitous urban data. In: Proceedings of the 24th ACM SIGSPATIAL, pp.1–4 (2016)
16. Tong, Y., Chen, Y., Zhou, Z., Lei, C., Lv, W.: The simpler the better: a unified approach to predicting original taxi demands based on large-scale online platforms. In: ACM SIGKDD, pp. 1653–1662 (2017)
17. Rose, Y., Li, Y., Shahabi, C., Demiryurek, U., Liu, Y.: Deep learning: a generic approach for extreme condition traffic forecasting. In: Chawla, N., Wang, W. (eds.) Proceedings of the 2017 SIAM International Conference on Data Mining, pp. 777–785. Society for industrial and applied mathematics, Philadelphia, PA (2017). https://doi.org/10.1137/1.9781611974973.87
18. Yao, H., et al.: Deep multi-view spatial-temporal network for taxi demand prediction. In: Proceedings of the AAAI, 32 (1) (2018)
19. Sun, Q., Liu, Y., Chua, T.S., Schiele, B.: Meta-transfer learning for few-shot learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 403–412 (2019)

20. Jiang, W., Huang, K., Geng, J., Deng, X.: Multi-Scale metric learning for few-shot learning. IEEE Trans, Circuits Syst. Video Technol. **31**(3), 1091–1102 (2021)
21. Ren, M., Liao, R., Fetaya, E., Zemel, R.: Incremental few-shot learning with attention attractor networks. Adv. NeurIPS **32**, 5275–5285 (2019)
22. Wei, Y., Zheng, Y., Yang, Q.: Transfer knowledge between cities. In: Proceedings of the 22nd ACM SIGKDD, pp. 1905–1914 (2016)
23. Yao, H., Tang, X., Wei, H., Zheng, G.: mRevisiting spatial-temporal similarity: a deep learning framework for traffic prediction, In: The 33rd AAAI Conference, pp. 5668–5675 (2019)
24. Shi, X., Chen, Z., Wang, H., Yeung, D., Wong, W.: Convolutional LSTM network: a machine learning approach for precipitation nowcasting. In: NeurIPS, pp. 802–810 (2015)
25. Yoon, S.W., Kim, D.Y., Seo, J., Moon, J.: Xtarnet: Learning to extract task-adaptive representation for incremental few-shot learning. In: ICML, pp. 10852–10860 (2020)
26. Xiang, L., Jin, X., Ding, G., Han, J., Li, L.: Incremental few-shot learning for pedestrian attribute recognition, In Proceedings of the 28th IJCAI 2019, pp. 3912–3918 (2019)
27. LIU, L., Zhou, T., Long, G., Jiang, J., Zhang, C.: Learning to Propagate for Graph Meta-Learning. In: Advances in Neural Information Processing Systems, 32, 1039–1050 (2019)
28. Liao, R., et al.: Reviving and improving recurrent back-propagation. In Proceedings of the 35th ICML 2018, Stockholmsmassan, Stockholm, Sweden, pp. 3088–3097 (2018)
29. Pineda, F.J.: Generalization of back propagation to recurrent and higher order neural networks. In: Anderson, D.Z., (ed.) Neural Information Processing Systems, Denver, Colorado, USA, American Institue of Physics, pp. 602–611 (1987)
30. Hyndman, R.J., Athanasopoulos, G. Forecasting: principles and practice. OTexts (2018)
31. Wang, L., Geng, X., Ma, X., Liu, F., Yang, Q.: Crowd flow prediction by deep spatio-temporal transfer learning. arXiv preprint arXiv:1802.00386 (2018)
32. Williams, R.J., Peng, J.: An efficient gradient-based algorithm for on-line training of recurrent network trajectories. Neural Comput. **2**(4), 490–501 (1990)
33. Sprechmann, P., Jayakumar, S.M., Rae, J.W., Pritzel.: Memory-based parameter adaptation. In: The 6th International Conference on Learning Representations, ICLR (2018)