Overcoming Catastrophic Forgetting in Continual Fine-Grained Urban Flow Inference

XOVEE XU, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China
TING ZHONG, School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, China
HAOYANG YU, School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, China
FAN ZHOU, School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, China
GOCE TRAJCEVSKI, Department of Electrical and Computer Engineering, Iowa State University, Ames, United States

Citywide fine-grained urban flow inference (FUFI) problem aims to infer the high-resolution flow maps from the coarse-grained ones, which plays an important role in sustainable and economic urban computing and intelligent traffic management. Previous models tackle this problem from spatial constraint, external factors, and memory cost. However, utilizing the new urban flow maps to calibrate the learned model is very challenging due to the "catastrophic forgetting" problem and is still under-explored. In this article, we make the first step in FUFI and present CUFAR—Continual Urban Flow inference with augmented Adaptive knowledge Replay—a novel framework for inferring the fine-grained citywide traffic flows. Specifically, (1) we design a spatial-temporal inference network that can extract better flow map features from both local and global levels; (2) then, we present an augmented adaptive knowledge replay (AKR) training algorithm to selectively replay the learned knowledge to facilitate the learning process of the model on new knowledge without forgetting. We apply several data augmentation techniques to improve the generalization capability of the learning model, gaining additional performance improvements. We also propose a knowledge discriminator to avoid the "negative replaying" issue introduced by noisy urban flow maps. Extensive experiments on two large-scale real-world FUFI datasets demonstrate that our proposed model consistently outperforms strong baselines and effectively mitigates the forgetting problem.

CCS Concepts: • **Information systems** → **Spatial-temporal systems**; *Sensor networks*; • **Applied computing** → *Transportation;*

This work was supported in part by the National Science Foundation under Grant SWIFT No. 2030249 and Kingland Foundation.

Authors' Contact Information: Xovee Xu, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, China; e-mail: xoveexu@gmail.com; Ting Zhong, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, China; e-mail: zhongting@uestc.edu.cn; Haoyang Yu, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, China; e-mail: haoyang.yu417@outlook.com; Fan Zhou (Corresponding author), School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, China; e-mail: fan.zhou@uestc.edu.cn; Goce Trajcevski, Department of Electrical and Computer Engineering, Iowa State University, Ames, Iowa, United States; e-mail: gocet25@iastate.edu.

ACM 2374-0353/2024/10-ART34
https://doi.org/10.1145/3660523

Additional Key Words and Phrases: fine-grained urban flow inference, urban computing, continual learning, spatio-temporal data mining

ACM Reference Format:
Xovee Xu, Ting Zhong, Haoyang Yu, Fan Zhou, and Goce Trajcevski. 2024. Overcoming Catastrophic Forgetting in Continual Fine-Grained Urban Flow Inference. *ACM Trans. Spatial Algorithms Syst.* 10, 4, Article 34 (October 2024), 26 pages. https://doi.org/10.1145/3660523

1 INTRODUCTION

Urban flow analysis, prediction, and inference are important applications of smart city development and urban computing. They have been used for traffic management and urban transportation planning [12, 20, 35, 39, 67], benefiting from the fast urbanization, vast data generated from IoT devices, and the new computing technologies in recent years [24, 56, 60, 62, 68, 75]. **Fine-grained urban flow inference (FUFI)**, which tries to infer the high-resolution flow map from the corresponding coarse-grained one, is proposed as an important step toward an environmentally friendly and sustainable urban traffic system.

UrbanFM [35] is the first work that formulates the FUFI problem and proposes a distributional upsampling module and an external factor fusing subnet for tackling the problem. Subsequent works improve UrbanFM from several important aspects, e.g., the spatial constraint, external factors, and memory cost. Specifically, FODE [73] and UrbanODE [71]—based on neural **ordinary differential equations (ODEs)**—are proposed to address the numerical instability problem in FUFI by an affine coupling layer and a pyramid attention network. MT-CSR [30] addresses the FUFI problem with incomplete urban flow map. DeepLGR [38] revisits the limitations of **convolutional neural network (CNN)** and tries to learn global spatial dependencies and local feature representations of the flow dynamics. UrbanPy [46] is the state-of-the-art FUFI model that extends UrbanFM by proposing a cascading strategy based on the pyramid architecture, a propose-and-correct component, and a new distribution loss.

Motivations. Despite the promising results achieved in prior works, several potential improvements are worth exploring. First, although existing works have designed global-local architectures such as pyramid mechanism [46, 71], global-local context module [38], and Transformer [37, 76] to learn the long-range dependencies between local regions at different granularity, they are still inefficient in modeling the spatial relations of urban flows while also considering the temporal flow dynamics. Second, prior methods learn each FUFI dataset in isolation and retrain the entire model with the newly obtained fine-grained urban flow maps, leaving the previous data unexploited. One straightforward solution is to train all the data at once. However, it has two drawbacks: (1) noise data may be introduced from the older flow maps that have different flow distributions; (2) the computation overhead becomes unaffordable as time goes on. Another solution is to continually fine-tune the trained model from previous data on the new data, which is efficient and feasible to take advantage of the learned urban flow knowledge. However, fine-tuning directly on the new data is very prone to the "catastrophic forgetting" problem—i.e., much of the learned knowledge is overridden upon learning the new knowledge—mainly due to the parameter-updating mechanism (e.g., back-propagation), resulting in the model less generalized and less robust. This is also a result of the "stability-plasticity" dilemma [9].

Present Work. We present **CUFAR: Continual Urban Flow inference with Adaptive knowledge Replay**, as a novel way of inferring the fine-grained flow map with the help of previously learned knowledge. Specifically, we design a simple yet effective inference network that extracts spatial-temporal flow map features from both local and global perspectives, enabling the model to infer more accurate flow distributions. Then, we propose a general **adaptive knowledge replay (AKR)** training algorithm to continually and selectively replay the old knowledge to facilitate the learning process of the model on new data while also overcoming the "catastrophic forgetting" problem. Moreover, we apply several data augmentation techniques on the replayed knowledge to improve the generalization capability of the learning model and gain additional performance improvements in the training phase. We further design an adaptive knowledge discriminator to measure the flow distribution difference before and after the knowledge replaying, which helps the model mitigate the "negative replaying" issue that may occur if noisy data are introduced.

Extensive experiments (including ablation study and visualization) on two large-scale real-world urban flow datasets demonstrate the effectiveness and robustness of CUFA over strong FUFI baselines. We have the following notable findings: (1) The proposed spatial-temporal inference network uniformly improved the FUFI performance on different types of urban flow datasets, which has a better capability for learning expressive flow map features. (2) The designed AKR training algorithm successfully alleviated the "catastrophic forgetting" problem in continual FUFI and, consequently, improved the inference performance. It is worth noting that all baselines equipped with AKR have better performance. (3) Interestingly, on TaxiBJ-P4 dataset, we both observed the "negative replaying" and "overfitting" if the proposed knowledge discriminator and AKR are removed, respectively. (4) Compared to the *joint* protocol, i.e., training all data at once, our approach is efficient and even outperforms *joint*. We speculate this deficiency of *joint* is due to the noisy samples introduced from previous data. The above findings verify our motivation and show that utilizing the prior knowledge (properly) to facilitate the learning process of current knowledge is a promising way toward a robust and sustainable urban transportation system. Our contributions are summarized as follows:

— To the best of our knowledge, we are the first to formulate continual FUFI problem and provide comprehensive analysis of recent FUFI methods on both traditional FUFI problem and continual FUFI problem.
— We propose a novel FUFI method, CUFAR, which can infer the fine-grained flow map with the help of previously learned knowledge. It consists of a better inference network and an adaptive knowledge replay training algorithm to overcome the "catastrophic forgetting" problem.
— We design a data augmentation powered adaptive knowledge discriminator to mitigate the "negative replaying" issue, which measures the distribution discrepancy before and after the knowledge replaying.
— We contribute a new continual FUFI dataset, TaxiNYC, which includes the taxi flows of New York City spanning over three years. Extensive evaluations between CUFAR and strong baselines demonstrate the effectiveness of our proposed model on overcoming the catastrophic forgetting problem and improving FUFI performance.

The rest of the article is organized as follows: Section 2 reviews the literature of urban flow prediction and inference. In Section 3, we formulate the FUFI problem. We then describe the framework and details of CUFAR in Section 4, which includes a global-local urban flow inference network and an adaptive knowledge replay algorithm. Section 5 presents experimental settings and results, including comprehensive ablation studies and case studies. At last, we conclude this article and point out future directions. This article is an extension of Reference [63], previously presented at the AAAI 2023 conference.

2 RELATED WORK

In this section, we first review continual learning methods and provide necessary background knowledge of them. Then, we introduce the literature of urban flow prediction and fine-grained urban flow inference.

2.1 Continual Learning

Overcoming the catastrophic forgetting in artificial neural networks is attracting much research attention [13] due to its significance for a dynamic system that has new data/tasks coming continuously. Continual learning aims to model a sequence of new tasks without forgetting the knowledge of past tasks, which is a promising direction towards sustainable and robust neural networks [45, 48, 72]. Early efforts on continual learning can be categorized into three types: (1) *Regularization-based* methods impose restrictions when learning a new task to mitigate catastrophic forgetting. They use specific loss functions to take these constraints on the parameter updating process and consolidate previously learned knowledge [27, 34, 45]. (2) *Parameter isolation* methods dedicate different model parameters to each task to prevent any possible forgetting [13]. When no constraints are applied to the size of the architecture, one can grow new branches for new tasks while freezing the parameters of the old task or provide a model copy to each task [1, 13, 50]. However, the above-mentioned two types of methods face several drawbacks: First, they often impose constraints when learning new tasks, which limits the generalizability of the model; second, they cannot utilize the knowledge from old tasks to improve new task performance. (3) *Replay-based* methods save the data of previous tasks in a memory buffer. When learning new tasks, they replay the samples from the buffer and then mitigate the catastrophic forgetting of previous tasks [6]. Compared to other continual methods, replay-based methods do not constrain the new task optimization to prevent the interference of previous tasks, and they are more suitable for FUFI. Motivated by replay-based methods, we design an adaptive knowledge replay algorithm for selectively replaying the knowledge from previous tasks and finally improving the FUFI performance on new tasks.

Continual learning is also closely related to data shift or concept drift [7, 16], which focuses on the changes over time in data streams. In many non-stationary real-world applications, the data distributions may change over time, and in consequence, cause unpredictability and performance degradation. In Reference [33], the authors proposed a method to forecast the data distribution evolution. A non-stationary Transformer is presented in Reference [42] to tackle the over-stationarization problem in time series forecasting. To adapt to unprecedented volatility caused by social events in human mobility modeling, an event-aware spatio-temporal network is designed in Reference [57].

2.2 Urban Flow Prediction

Traffic flow analytics and predictions are widely used in intelligent transportation systems and are powered by the advances in data mining and deep learning techniques, which play an important role in traffic management [4, 5], human trajectory modeling [19, 70, 74], fine and urban planning [55]. One of the most attractive research interests in traffic flow analytics is the urban/traffic flow prediction that focuses on predicting the future traffic flows given historical flow observations [23, 25]. Traditional urban flow prediction methods mainly used time-series-based traffic flows collected from monitors to predict the future urban flows, e.g., statistics-based prediction model ARIMA. However, these statistics methods cannot model spatio-temporal features [23, 29]. More recently, deep learning–based methods are proposed to learn urban flow patterns from various perspectives (e.g., spatial and temporal dependencies, external factors). Recurrent and convolutional neural networks are widely used to address this problem. For example, ST-ResNet consists of a residual neural network and residual convolutional units, aiming to jointly predict inflow and outflow of crowds in city grid cells [66]. AutoST leverages optional convolution operations to extract multi-range spatio-temporal dependencies and uses learnable skip connections to fuse multimodal features [32]. In Reference [36], the authors proposed STRN, which designs a global relation module to efficiently learn global spatial dependencies and a meta learner for learning meta knowledge to improve the model performance. With the developments of **graph neural networks (GNNs)**, researchers found that GNNs are ideally suited to traffic flow prediction due to their capabilities to model non-Euclidean graph structures and capture spatial dependencies (e.g., road intersections serve as nodes and road connections serve as edges, which naturally resembles a road network) [17, 23]. Our problem differs from urban flow prediction in that it seeks to infer fine-grained urban flows by giving the coarse-grained ones, rather than forecasting future urban flows.

2.3 Fine-grained Urban Flow Inference

Acquiring high-resolution urban flows citywide requires expensive maintenance and operational costs for smart-city systems and applications [58, 59]. FUFI is proposed to reduce the number of deployed surveillance devices and sensors while keeping the original data granularity unchanged [35]. FUFI problem is beneficial for sustainable traffic management system. Despite its similarity to **single image super-resolution (SISR)** [35, 73], FUFI possesses the following unique challenges: (1) *Spatial constraint.* Different from SISR, FUFI problem has to obey the spatial constraint that the sum of the flow volumes in a coarse-grained flow cell is strictly equal to the sum of flow volumes in the corresponding region in fine-grained flow map. (2) *External factors.* Traffic flows are greatly affected by many external factors, such as weather conditions, temperature, date and time, and so on. These factors are important for inferring accurate fine-grained flow maps. (3) *Spatio-temporal continuity.* Urban flow maps usually have similarities in continuous time systems, e.g., the road structures are identical, but the flow volumes are continually changing. Generally, SISR does not has such continuous features. It can achieve better performance by learning the continuous spatio-temporal features in FUFI. UrbanFM is the first work to formulate and solve the FUFI problem. It proposed N^2-Normalization to handle the spatial constraint and proposed an external factor fusing subnet to process external factors. Subsequent works improve UrbanFM from several important aspects, e.g., the spatial constraint, external factors, and memory cost. Specifically, to explicitly address the numerical instability issue in FUFI, FODE [73] and UrbanODE [71] utilized **neural ordinary differential equations (NODE)** and proposed an affine coupling layer. MT-CSR [30] addressed the FUFI problem in which the coarse-grained flow maps are sparse and incomplete. DeepLGR [38] revisited the limitations of CNNs in solving urban flow–related tasks and proposed to learn local feature representations and global spatial dependencies of the flow dynamics. UrbanPy [46] is the state-of-the-art FUFI model that builds on UrbanFM by adding a propose-and-correct component, a novel distribution loss, and a pyramid-based cascade architecture. However, existing FUFI methods treat each flow map independently. When deploying such methods in practical situations, they are prone to face the problem of "catastrophic forgetting" when using new urban flow maps to calibrate the learned model, resulting in degenerated performance or otherwise a heavy and inefficient *joint* model trained on all urban flow maps.

3 PROBLEM FORMULATION

In this section, we formally define the problem of FUFI. The mathematical notations frequently used throughout the article are summarized in Table 1. In FUFI, we aim to infer the citywide fine-grained flow map from the coarse-grained one. Given a city of interest, we divide the city's map M into grid-cells. For each cell, we record its traffic flows $x_{ij} \in \mathbb{R}_+$ every τ minutes. The overall

Table 1. Notations and Descriptions

Symbol	Description
x_{ij}	Traffic flows in a grid of map
H, W	The height and width of coarse-grained maps
$\mathbf{X}_{cg}, \mathbf{X}_{fg}$	Coarse-grained map and fine-grained map
N	Upscaling factor
$\mathbf{H}_{cg}^{\text{local}}, \mathbf{H}_{fg}^{\text{local}}$	Coarse- and fine-grained feature maps of local regions
$\mathbf{H}_{fg}^{\text{global}}, \mathbf{H}_{fg}$	Fine-grained feature maps of global region, the output of spatial relation extractor
$\mathcal{M}, \mathcal{M}_{\text{sub}}$	Memory buffer and sub-memory buffer
S	Size of memory buffer $\mathcal{M}$
$B_{\mathcal{M}}$	Size of mini-batch of the buffer
$\mathbf{X}_{cg}^{\text{merge}}, \mathbf{X}_{fg}^{\text{merge}}$	Merged coarse- and fine-grained maps
$\mathbf{X}_{cg}^{\text{DA}}, \mathbf{X}_{fg}^{\text{DA}}$	Data augmented coarse- and fine-grained maps

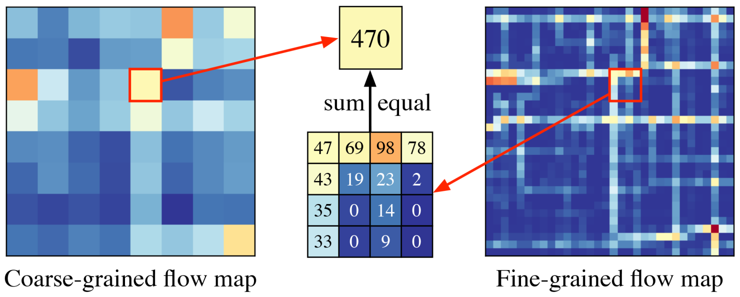

Fig. 1. Spatial constraint between coarse- and fine-grained flow maps in a local area of Beijing city.

traffic flows of M are denoted as $\mathbf{X}$. Following existing works [35, 46], the FUFI problem and its spatial constraint are defined as:

Definition 3.1 (Fine-grained Urban Flow Inference). Given a coarse-grained map $\mathbf{X}_{cg} \in \mathbb{R}_{+}^{H \times W}$, the FUFI problem is to infer the corresponding fine-grained flow map $\mathbf{X}_{fg} \in \mathbb{R}_{+}^{NH \times NW}$; here, N is an upscaling factor.

Definition 3.2 (Spatial Constraint). Different from image super-resolution, FUFI problem has to obey a spatial constraint that the cell flow x_{ij} of the coarse-grained map is strictly equal to the sum of flows in the corresponding $N \times N$ cells of the fine-grained map, i.e.:

$$x_{ij,cg} = \sum_{i'j'} x_{i'j',fg} \quad s.t. \left\lfloor \frac{i'}{N} \right\rfloor = i, \left\lfloor \frac{j'}{N} \right\rfloor = j, \tag{1}$$

where $i = 1, 2, \ldots, H$ and $j = 1, 2, \ldots, W$. An illustration of the spatial constraint is depicted in Figure 1.

In traditional FUFI problem settings [31, 35], the learning model is retrained entirely every time new urban data comes in, which belongs to the offline learning paradigm. Online learning, on the contrary, dynamically and sequentially learns new data patterns, enabling an efficient and sustainable prediction model that is also the target of the FUFI problem. However, online learning algorithms such as continual learning and fine-tuning may face the challenge of catastrophic forgetting when adapting to new data/tasks.[1] In this work, we study the continual FUFI problem, i.e.,

[1]When there is no ambiguity, we interchangeably use data and task throughout the article.

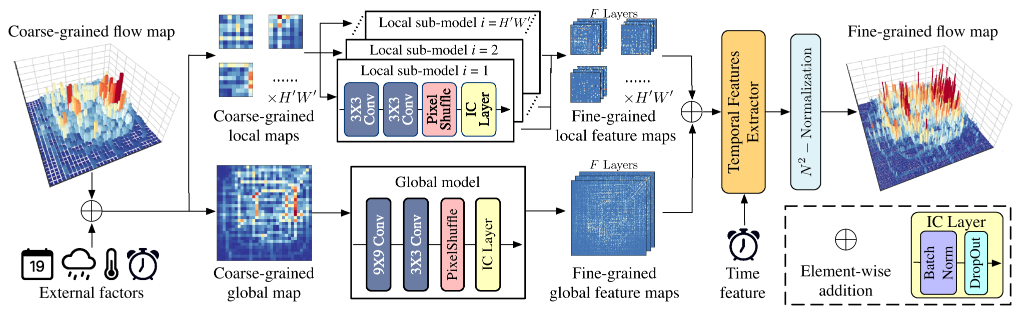

Fig. 2. The framework of the proposed inference network. It first extracts global-local map features from the coarse-grained flow map along with external factors such as weather and date. Then, extracted flow map features are combined with temporal features and an N^2-Normalization layer to infer the final fine-grained flow map.

given a sequence of urban flow datasets ordered over time, we learn new knowledge with the help of old knowledge, but without forgetting the old knowledge.

4 METHODOLOGY

We now illustrate the CUFAR methodology, which contains two critical components: (1) an inference network consisting of two feature extractors for learning the spatial-temporal relations of urban flow maps from both global and local levels; (2) a specifically designed continual algorithm that combines the experience replay strategy with an adaptive knowledge discriminator. Moreover, we apply data augmentation techniques after replaying old knowledge. The framework of the inference network is depicted in Figure 2, and the continual algorithm is described in Algorithm 1.

4.1 Spatial-temporal Inference Network

4.1.1 Spatial Relation Extraction. Urban road topological structures and the corresponding temporal flow dynamics are extremely complex and cannot be parsed with simple rules. With the help of CNNs, previous FUFI methods [35, 71, 73] learn global feature maps with shared kernel weights to infer the fine-grained flow map. Moreover, they often adopted complex architecture design (e.g., overstacking residual layer, pyramid attentions, cascade architecture) and may lead to overfitting, low calculating efficiency, and unstable gradient computation problems in the training phase. As a result, these methods are inefficient to model the local area flow dynamics. To address this hurdle, we design a spatial relation extraction module to partition the flow map into smaller local regions and use several standalone sub-models to separately infer the upscaled local maps. Specifically, for $H'W'$ sub-regions, each sub-model consists of two convolution layers followed by a PixelShuffle layer and an **independent component (IC)** layer. The first convolution layer has F filters (3×3) and the second has FN^2 filters; recall that N is the upscale factor. The PixelShuffle layer upscales the coarse-grained feature maps $\mathbf{H}_{cg}^{\text{local},i} \in \mathbb{R}^{\frac{H}{H'} \times \frac{W}{W'} \times FN^2}$ to fine-grained ones $\mathbf{H}_{fg}^{\text{local},i} \in \mathbb{R}^{\frac{NH}{H'} \times \frac{NW}{W'} \times F}$. The IC layer [11] is composed of a batch normalization layer and a dropout layer. It can whiten the mutual information and correlation coefficients between network neurons and accelerate the speed of convergence. The global model is similar to the local sub-model but has a larger filter size (9×9) in the first convolution layer. Its output is the global fine-grained feature maps $\mathbf{H}_{fg}^{\text{global}} \in \mathbb{R}^{NH \times NW \times F}$. We then restore the undivided feature maps $\mathbf{H}_{fg}^{\text{local}} \in \mathbb{R}^{NH \times NW \times F}$ from local feature maps. At last, we concatenate the obtained feature maps as the input of the temporal feature extractor: $\mathbf{H}_{fg} = [\mathbf{H}_{fg}^{\text{global}}; \mathbf{H}_{fg}^{\text{local}}] \in \mathbb{R}^{NH \times NW \times 2F}$.

4.1.2 Temporal Features Extraction. Various external conditions such as *weather* and *date* have an influence on the distribution of citywide urban flows. Among them, time span is closely related to flow volumes but largely ignored or dealt with as a normal external condition. To remedy this, we propose a temporal feature extraction module based on convolutional sequences to capture the influence of time factor on the traffic flow distribution. Specifically, we build K independent convolutional layers (F filters, 3×3), each account for a specific time span in a day. The weights of these layers are non-shared. Then, the fine-grained feature maps $\mathbf{H}_{fg}$ are fed into the corresponding convolutional layer followed by an N^2-Normalization layer to obtain the desired fine-grained flow map $\widetilde{\mathbf{X}}_{fg} \in \mathbb{R}^{NH \times NW}$.

4.1.3 External Factors and N^2-normalization. As same as the one in UrbanFM [35], we adopt several embedding layers to transform the external factors (except the time span factor) into low-dimensional vectors and then use dense layers to reshape the vectors into a coarse-grained feature map. Thereby, the input of our model is the concatenation of the coarse-grained flow map, external feature map, and time span feature map. To obey the spatial constraint required by FUFI problem, existing methods often adopt the N^2-Normalization as model's last layer instead of adding new losses. In CUFAR, we also use this normalization trick. The details of the external factors and N^2-Normalization can be found in UrbanFM [35].

4.1.4 Optimization. The training objective of CUFAR is the widely used **mean squared error (MSE)** between the inferred flow map and the ground truth:

$$\mathcal{L} = ||\widetilde{\mathbf{X}}_{fg} - \mathbf{X}_{fg}||^2. \tag{2}$$

Next, we illustrate how we adaptively and continually learn new knowledge without forgetting the old knowledge.

4.2 Adaptive Knowledge Replay

When we use the newly obtained urban flow maps to calibrate the trained model in FUFI (we denote this process as learning on a task), e.g., fine-tuning on the new data, the learned knowledge tends to be overridden by the new knowledge due to the parameter-updating mechanism (e.g., back-propagation), resulting in catastrophic forgetting that lowers the inference performance. In this work, we propose an **adaptive knowledge replay (AKR)** training method that continually and selectively replays the old knowledge to help the learning process of the new task. AKR consists of a memory buffer and an adaptive knowledge discriminator. The framework of AKR is depicted in Figure 3.

4.2.1 Selective Memory Buffer. Prior works [6, 43, 49] show that one effective way to overcome catastrophic forgetting is the experience replay, which saves the old data in a memory buffer and does not constrain the optimization process. Inspired by experience replay, we use a memory buffer $\mathcal{M}$ to reserve the learned urban flow maps from previous tasks, which has a max size S. In every training iteration, we randomly sample a buffer mini-batch $B_{\mathcal{M}}$ from $\mathcal{M}$. The sampled $B_{\mathcal{M}}$ is then merged with the original training batch B. Different from existing experience replay-based methods [6, 49], our designed selective memory buffer (1) does not retrospect current training data and (2) contains a sub-memory buffer $\mathcal{M}_{\text{sub}}$ of size $S/2$ for storing recent data on the new task. Specifically, during the training on the first task, buffer $\mathcal{M}$ is filled and then updated by the reservoir sampling algorithm [54]. During the training of subsequent tasks, the sub-memory buffer $\mathcal{M}_{\text{sub}}$ replaces the role of the $\mathcal{M}$, i.e., we update $\mathcal{M}_{\text{sub}}$ with the new data and replay the old data from $\mathcal{M}$. Every time we train on new task, half of the data in $\mathcal{M}$ is randomly replaced by the data (from the last task) in $\mathcal{M}_{\text{sub}}$. Therefore, the buffer $\mathcal{M}$ has more recent samples.

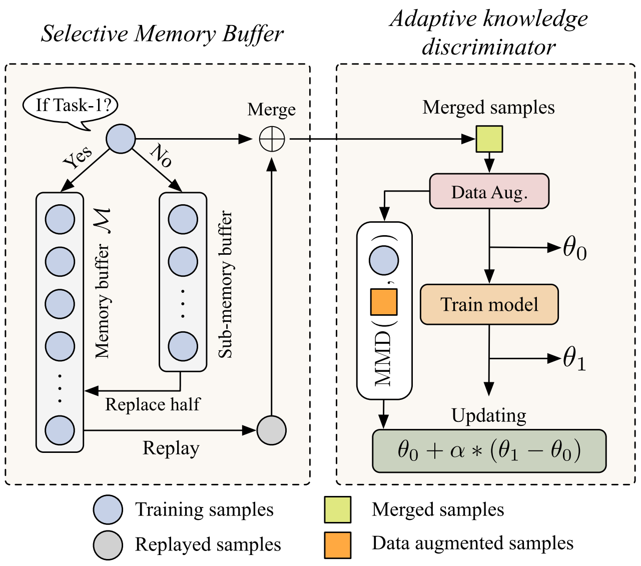

Fig. 3. Framework of adaptive knowledge replay.

4.2.2 Data Augmentation. In our prior work [63], when we train models with AKR, we first replay past data from the memory buffer $\mathcal{M}$ and merge with current training data. We then calculate the discrepancy between merged data and original training data, which is used to alleviate the influence of out-of-distribution noises when updating the model. During this process, we found that utilizing **data augmentation (DA)** techniques on replayed data can improve the generalization capability of the model and gain additional performance improvements. DA is proven as one of the most effective ways to address over-fitting problem and boost prediction accuracy. It acts like a regularizer and only brings small computational costs in the training phase [61]. Specifically, we first merge the replayed data with current training data and get merged samples $\mathbf{X}_{cg}^{\text{merge}}$ and $\mathbf{X}_{fg}^{\text{merge}}$. Then, we augment merged samples $\mathbf{X}_{cg}^{\text{merge}}$ and $\mathbf{X}_{fg}^{\text{merge}}$ by randomly selecting one of the following DA techniques:

— **Cutout** [14] is a regional dropout method, which randomly cuts regions in images to improve the generalization performance of CNN-based models [64, 69]. In this work, for each pixel in the flow map, we set the removing probability as 0.001.
— **Mixup** [65] extends sample distributions by incorporating prior knowledge into the training sample via mixing two samples. It can alleviate the overfitting problem in SISR and enhance the robustness of neural networks in the training phase [18, 65]. We blend the merged sample and the random permutation sample of itself in our work.
— **CutMix** [64] extends Cutout by replacing the removed regions with a patch from another image, which does not drop information of the training sample [8, 64]. Specifically, we randomly select a patch of the flow map and then blend the corresponding region of the merged sample and the random permutation sample of itself.
— **CutMixup** [61] combines both Mixup and Cutmix; it first blends two samples and then replaces random region of the training sample. CutMixup benefits from minimizing the boundary effect and the ratio of the mixed regions, achieving better performance compared to other DA techniques. We implement CutMixup based on the simple combination of Mixup and Cutmix.

ALGORITHM 1: Adaptive Knowledge Replay (AKR).

```
Require: A sequence of tasks $\{T_1, T_2, \ldots\}$ containing coarse- and fine-grained flow maps $\mathbf{X}_{cg}$ and $\mathbf{X}_{fg}$,
    buffer $\mathcal{M}$ and $\mathcal{M}_{\text{sub}}$.
 1: $\mathcal{M} \leftarrow \varnothing, \mathcal{M}_{\text{sub}} \leftarrow \varnothing$;
 2: for $T_i$ in $\{T_1, T_2, \ldots\}$ do
 3:     if $T_1$ then
 4:         for sampled mini-batch $B = \{(\mathbf{X}_{cg}^j, \mathbf{X}_{fg}^j)\}_{j=1}^{|B|}$ do
 5:             Fill $\mathcal{M}$ using the reservoir sampling algorithm;
 6:             for $\mathbf{X}_{cg}^j \in B$ do
 7:                 $\widetilde{\mathbf{X}}_{fg}^j \leftarrow \text{Model}(\mathbf{X}_{cg}^j)$;
 8:             end for
 9:             $\theta \leftarrow \text{MSELoss}(\{\widetilde{\mathbf{X}}_{fg}^j, \mathbf{X}_{fg}^j\}_{j=1}^{|B|})$;        ▷ Updating
10:         end for
11:     else                                                  ▷ Starting to replay knowledge
12:         if $\mathcal{M}_{\text{sub}} \neq \varnothing$ then
13:             Replace half of $\mathcal{M}$ with $\mathcal{M}_{\text{sub}}$ and set $\mathcal{M}_{\text{sub}} = \varnothing$;
14:         end if
15:         for sampled mini-batch $B = \{(\mathbf{X}_{cg}^{\text{ori},j}, \mathbf{X}_{fg}^{\text{ori},j})\}_{j=1}^{|B|}$ do
16:             Fill $\mathcal{M}_{\text{sub}}$ using the reservoir sampling algorithm;
17:             Sample $B_{\mathcal{M}}$ from the buffer $\mathcal{M}$;
18:             $\{(\mathbf{X}_{cg}^{\text{merge},j}, \mathbf{X}_{fg}^{\text{merge},j})\}_j \leftarrow B \cup B_{\mathcal{M}}$;
19:             $B^{\text{DA}} = \text{Data Augmentation}(\{(\mathbf{X}_{cg}^{\text{merge},j}, \mathbf{X}_{fg}^{\text{merge},j})\}_j)$
20:             $\alpha \leftarrow d_{\text{MMD}}^2(B^{\text{DA}}, B), \theta_0 \leftarrow \theta$;
21:             for $\mathbf{X}_{cg}^{\text{DA},j} \in B^{\text{DA}}$ do
22:                 $\widetilde{\mathbf{X}}_{fg}^{\text{DA},j} = \text{Model}(\mathbf{X}_{cg}^{\text{DA},j})$;
23:             end for
24:             $\theta_1 \leftarrow \text{MSELoss}(\{\widetilde{\mathbf{X}}_{fg}^{\text{DA},j}, \mathbf{X}_{fg}^{\text{DA},j}\}_{j=1}^{B^{\text{DA}}})$;
25:             $\theta \leftarrow \theta_0 + \alpha * (\theta_1 - \theta_0)$;                        ▷ Updating
26:         end for
27:     end if
28: end for
```

Researchers have proven that randomly applying several DA methods is better than solely applying one of them [61]. Inspired by this insight, we apply the above-mentioned DA methods on merged samples to get $\mathbf{X}_{cg}^{\text{DA}}$ and $\mathbf{X}_{fg}^{\text{DA}}$. The augmented samples are then fed into the training model. Before that, we will calculate the domain distance using maximum mean discrepancy, which we will detail in the following subsection.

4.2.3 Adaptive Knowledge Discriminator. When replaying past data from the memory buffer $\mathcal{M}$, if the distribution of replayed urban flow maps greatly differs from the original distribution, then "negative replaying" may occur. To mitigate this deficiency, we further introduce a **maximum mean discrepancy (MMD)** [21] to measure the similarity between the replayed distribution and original distribution. MMD is often used in domain adaptation and transfer learning to measure the domain distance and constrain the representation space [2, 3, 40]. We use MMD as a distance discriminator to prevent our model being affected by the out-of-distribution flow maps (e.g., unusual traffic flow distribution due to road accidents or lockdowns). Given two distributions—in our case, the replayed samples $\mathbf{X}_{cg}^{\text{DA}}$ from data augmented batch and the samples $\mathbf{X}_{cg}^{\text{ori}}$ from original batch—the MMD distance is defined as:

$$d_{\mathrm{MMD}}(\mathbf{X}_{cg}^{\mathrm{DA}}, \mathbf{X}_{cg}^{\mathrm{ori}}) = \sup_{f \in \mathcal{H}} \left(\mathrm{E}[f(P)] - \mathrm{E}[f(Q)] \right), \tag{3}$$

where $P \sim \mathbf{X}_{cg}^{\mathrm{DA}}$, $Q \sim \mathbf{X}_{cg}^{\mathrm{ori}}$, and $\mathcal{H}$ indicates **reproducing kernel Hilbert space (RKHS)**. We define two kernel embeddings $\mu_p := E[k(\cdot, P)]$ and $\mu_q := E[k(\cdot, Q)]$; here, $k(\cdot,\cdot) \in \mathcal{H}$. Then, the MMD distance can be derived as:

$$\begin{aligned} d_{\mathrm{MMD}}^2(\mathbf{X}_{cg}^{\mathrm{DA}}, \mathbf{X}_{cg}^{\mathrm{ori}}) &= \left[\sup_{\|f\|_{\mathcal{H}} \leqslant 1} \left\langle \mu_p - \mu_q, f \right\rangle_{\mathcal{H}} \right]^2 \\ &\leqslant \sup_{\|f\|_{\mathcal{H}} \leqslant 1} \left\| \mu_p - \mu_q \right\|_{\mathcal{H}}^2 \|f\|_{\mathcal{H}}^2 = \left\| \mu_p - \mu_q \right\|_{\mathcal{H}}^2 . \end{aligned} \tag{4}$$

Since the above equation cannot be computed directly, we expand the kernel function and draw i.i.d. samples $P = \{p_i\}_{i=1}^m$ from $\mathbf{X}_{cg}^{\mathrm{DA}}$ and $Q = \{q_j\}_{j=1}^n$ from $\mathbf{X}_{cg}^{\mathrm{ori}}$. Then, the squared MMD can be estimated as follows [3, 21]:

$$\begin{aligned} d_{\mathrm{MMD}}^2(P, Q) =& \frac{1}{m(m-1)} \sum_{i=1}^{m} \sum_{j \neq i}^{m} k\left(p_i, p_j\right) \\ &+ \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i}^{n} k\left(q_i, q_j\right) - \frac{2}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} k\left(p_i, q_j\right). \end{aligned} \tag{5}$$

We then introduce three kernel matrices, $\mathbf{K}_{S,S}$, $\mathbf{K}_{T,T}$, and $\mathbf{K}_{S,T}$, which represent the matrices of source domain (P), target domain (Q), and cross-domain data defined by $k(\cdot,\cdot)$ [47], respectively. Then, we can rewrite Equation (5) as:

$$d_{\mathrm{MMD}}^2(P, Q) = \mathrm{tr}\left(\begin{bmatrix} \mathbf{K}_{S,S} & \mathbf{K}_{S,T} \\ \mathbf{K}_{T,S} & \mathbf{K}_{T,T} \end{bmatrix} \mathbf{L} \right), \tag{6}$$

where

$$(\mathbf{L})_{ij} = \begin{cases} \frac{1}{mm}, & p_i, p_j \in P \\ \frac{1}{nn}, & p_i, p_j \in Q \\ \frac{-1}{mn}, & \text{otherwise.} \end{cases} \tag{7}$$

The range of d_{MMD}^2 is $[0, +\infty]$. We then normalize the distance into $(0, 1]$ by:

$$\alpha = 2 - \frac{2}{1 + e^{-d_{\mathrm{MMD}}^2}}. \tag{8}$$

The larger the α, the higher the similarity, and $\alpha = 1$ indicates that the two sample groups are identical. When replaying the learned knowledge on new tasks, for each training iteration, let θ_0 be the initial weights, θ_1 be the trained weights, the final model weights θ are updated as:

$$\theta = \theta_0 + \alpha * (\theta_1 - \theta_0). \tag{9}$$

The overall training process of the proposed adaptive knowledge replay is sketched in Algorithm 1.

5 EXPERIMENTS

We now evaluate the effectiveness of our proposed spatial-temporal inference network and adaptive knowledge replay on continual FUFI problem. Source code and datasets are publicly released at https://github.com/PattonYu/CUFAR

Table 2. Dataset Statistics

Dataset	TaxiBJ	TaxiNYC
P1 time range	07/01/2013–10/31/2013	07/1/2014–11/30/2014
P2 time range	02/01/2014–06/30/2014	01/1/2015–05/30/2015
P3 time range	03/01/2015–06/30/2015	07/1/2015–11/30/2015
P4 time range	11/01/2015–03/31/2016	01/1/2016–05/30/2016
Time interval	30 minutes	1 hour
$\mathbf{X}_{cg}$ size	32×32	16×16
$\mathbf{X}_{fg}$ size	128×128	64×64
Number of maps	P1: 3,060, P2: 3,559	P1: 3,672, P2: 3,624
	P3: 3,492, P4: 4,244	P3: 3,672, P4: 3,648
Avg. flow volume	P1: 12.438, P2: 14.809	P1: 6.207, P2: 6.159
	P3: 16.309, P4: 13.078	P3: 5.396, P4: 5.520
Weather conditions	17 types	17 types
Temperature	[−24.6°C, 41.0°C]	[−6.0°C, 33.0°C]
Wind speed	[0 mph, 48.6 mph]	[0 mph, 18.6 mph]

5.1 Data Description

Experiments are conducted on two real-world large-scale datasets: TaxiBJ and TaxiNYC.

- **TaxiBJ [35].** This dataset consists of four sub-datasets collected continuously for four years (2013 to 2016) in Beijing, which contains about 3.3 billion real-world trajectory data points. We denote four sub-datasets as TaxiBJ Task-1 to Task-4. In this dataset, the size of coarse-grained map is 32×32, the size of fine-grained map is 128×128, and upscaling factor N is 4.
- **TaxiNYC.** This is a new FUFI dataset that we collected from Taxi & Limousine Commission of NYC.[2] We select data from 2014 to 2016 with about 0.3 billion trajectory data points in total. We divide the whole dataset into four separate sub-datasets and each sub-dataset consists of five months' data, the size of coarse-grained map is 16×16, the size of fine-grained map is 64×64, and upscaling factor N is 4. Following the settings of TaxiBJ, we also consider the external factors in TaxiNYC, the meteorology data (e.g., the weather conditions, temperature, and wind speed) are collected from website.[3]

All datasets are associated with external factors that may affect urban flow distributions. They include: temperature, wind speed, weather, holiday, weekend, day of week, and hour of day. The weather conditions include: sunny, cloudy, overcast, rainy, sprinkle, moderate rain, heavy rain, rainstorm, thunderstorm, freezing rain, snowy, light snow, moderate snow, heavy snow, foggy, sandstorm, dusty. Table 2 shows the statistics of TaxiBJ and TaxiNYC.

5.2 Baselines

We compare CUFAR with the following 10 methods on FUFI problem. They include 5 single-image super-resolution approaches: SRCNN [15], VDSR [26], ESPCN [51], SRResNet [28], and DeepSD [53]; and 5 state-of-the-art FUFI approaches: UrbanFM [35], DeepLGR [38], FODE [73], UrbanODE [71], and UrbanPy [46]. The details of the 10 baselines are as follows:

- **SRCNN [15]:** is the first successful implementation of **convolutional neural networks (CNN)** in SR problem, which directly learns the mapping between low-resolution and high-resolution images and shows the promising results compared to the traditional SR methods.

[2] https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page
[3] https://en.tutiempo.net

- **VDSR** [26]**:** realizes real-time video super-resolution by leveraging VGG-net [52] as the backbone and achieves deep network structure with 20 depth, which also suggests that the large depth will show a significant improvement in accuracy.
- **ESPCN** [51]**:** introduces a sub-pixel convolutional layer to aggregate features from low-resolution space, which effectively replaces the bicubic filter in SR with a more complex upsampling filter trained specifically for each feature map while also improving computational efficiency.
- **SRResNet** [28]**:** is a ResNet-based [22] SR method, which introduces the deep residual network as the backbone and recovers photo-realistic textures from a large number of down-sampled images on a public benchmark.
- **DeepSD** [53]**:** is inspired by the adaptation SR image processing techniques to statistical downscaling, which enhances SRCNN with multi-scale input channels to maximize predictability in statistical downscaling and considers spatio-temporal nature of the climate system.
- **ESRT** [44]**:** targets the challenges of using Transformer architectures in the realm of computer vision, particularly the high computational cost and extensive GPU memory requirements. It is a hybrid model consisting of a CNN-based backbone and a Transformer-based backbone.
- **UrbanFM** [35]**:** is the first work in solving FUFI problem and stacks residual structure network as the backbone. It considers both spatial constraints and external conditions in flow map inference and achieves impressive results compared to traditional SISR methods.
- **DeepLGR** [38]**:** extends UrbanFM with a global-local feature extraction module and revisits the limitation of CNNs in learning the dynamic urban flow.
- **FODE** [73]**:** addresses numerically unstable gradient computation and huge memory cost problems with neural **ordinary differential equations (ODEs)**. It also introduces an augmented distributional upsampling layer to enhance the influence of external factors on inference.
- **UrbanODE** [71]**:** introduces a pyramid attention network to infer high-quality flow maps based on neural ODEs. Besides, UrbanODE is a memory-efficient model and provides a balance between accurately inferring urban flow maps and efficient computation.
- **UrbanPy** [46]**:** extends UrbanFM by decomposing the original task into multiple sub-tasks to address the insufficiency problem. By using cascading strategy based on pyramid architecture, UrbanPy achieves state-of-the-art results in inferring flow maps at higher upscaling rates.

5.3 Evaluation Metrics

Following previous FUFI methods [35, 46, 73], we use commonly used regression metrics including **mean squared error (MSE), mean absolute error (MAE)**, and **mean absolute percentage error (MAPE)**:

$$\mathrm{MSE} = \left\| \widetilde{\mathbf{X}}_{fg} - \mathbf{X}_{fg} \right\|^2, \tag{10}$$

$$\mathrm{MAE} = \left\| \widetilde{\mathbf{X}}_{fg} - \mathbf{X}_{fg} \right\|, \tag{11}$$

$$\mathrm{MAPE} = \left\| \frac{\widetilde{\mathbf{X}}_{fg} - \mathbf{X}_{fg}}{\mathbf{X}_{fg}} \right\|, \tag{12}$$

where $\widetilde{\mathbf{X}}_{fg}$ is the inferred fine-grained flow map and $\mathbf{X}_{fg}$ is the ground truth.

Table 3. Hyper-parameters in CUFAR

Parameter	**Search space**
Batch size B	16
Buffer mini-batch $B_{\mathcal{M}}$	$\{1, \mathbf{2}, 4, 6, 8\}$
Filter size F	$\{32, 64, \mathbf{128}\}$
Learning rate	$1e^{-4}$
Memory buffer size S	$\{100, 200, 500, \mathbf{1{,}000}, 1500\}$
Temporal conv layers K	$\{15, 24\}$
Upscaling factor N	4

Selected values are marked in **bold**.

5.4 Hyper-parameter and Implementation

All datasets are divided into training, validation, and test set in a ratio of 2:1:1. The hyper-parameters used in CUFAR and their corresponding values or search spaces are listed in Table 3. Hyper-parameters of baselines are set to their preferred values indicated in their papers. Al experiments are conducted on RTX 3090 with PyTorch. The optimizer is Adam, learning rate is $1e^{-4}$, filter size F is 128, temporal conv layers K is 15 (hourly from 7 AM to 10 PM) for TaxiBJ and 24 (the whole day) for TaxiNYC, the division H' and W' of sub-regions are 4, memory buffer size S is 1,000, batch B and $B_{\mathcal{M}}$'s sizes are 16 and 2, respectively. We use group convolution to implement the spatial relation extractor for multi-threaded computing; the group is $H'W' = 16$. Experiment results we report are the best of five runs of all methods. We note that we omit the results of SISR baselines on TaxiNYC, since their performances are significantly worse than FUFI baselines.

5.5 Training Protocols

We use four training protocols:

— *Single-task*: learns each task in isolation.
— *Joint*: learns previous tasks and new task at once, which costs a lot of computations when there are many tasks.
— *Fine-tune*: learns new task with previous task's trained model as initial, but is prone to catastrophic forgetting problem.
— *continual*: learns new task with our designed AKR algorithm to prevent forgetting problem.

For each dataset, it is divided chronologically into four sub-datasets, each representing a task (from P1 to P4). In real-world scenarios, practitioners may periodically update the trained model by incorporating newly collected data (a new task). The frequency of model updating may vary, ranging from weekly, monthly, to yearly, depending on specific monitoring requirements. *Fine-tune* and *continual* protocols learn new task with the help of previous task(s), but do not require retraining on them.

5.6 Evaluation Results

5.6.1 Spatial-temporal Inference Network. Table 4 shows the inference performance of our model and 10 baselines on four tasks of each dataset using the *single-task* protocol, i.e., we infer the fine-grained urban flow without fine-tuning or knowledge-replaying on the new task. We can observe CUFAR achieves the best results through all metrics on all tasks, which verifies the effectiveness of our designed spatial-temporal inference network.

5.6.2 Adaptive Knowledge Replay. Next, we evaluate the proposed AKR training algorithm and see if we can mitigate catastrophic forgetting while also improving the inference performance.

Table 4. Inference Performance between CUFAR and Baselines on TaxiBJ and TaxiNYC Datasets when Using the *single-task* Protocol

	Method	Task-1			Task-2			Task-3			Task-4		
		MSE	MAE	MAPE	MSE	MAE	MAPE	MSE	MAE	MAPE	MSE	MAE	MAPE
TaxiBJ	SRCNN	18.464	2.491	0.714	21.270	2.681	0.689	23.184	2.829	0.727	14.730	2.289	0.665
	ESPCN	17.690	2.497	0.732	20.875	2.727	0.732	22.505	2.862	0.773	13.898	2.228	0.711
	VDSR	17.297	2.213	0.467	21.031	2.498	0.486	22.372	2.548	0.461	13.351	1.978	0.411
	DeepSD	17.272	2.368	0.614	20.738	2.612	0.621	22.014	2.739	0.682	15.031	2.297	0.652
	SRResNet	17.338	2.457	0.713	20.466	2.660	0.688	21.996	2.775	0.717	13.446	2.189	0.637
	ESRT	17.252	2.210	0.552	20.472	2.664	0.688	22.001	2.751	0.699	13.521	2.210	0.649
	UrbanFM	16.372	2.066	0.335	19.548	2.284	0.328	21.243	2.398	0.336	12.744	1.850	0.311
	DeepLGR	17.125	2.103	0.339	21.217	2.386	0.350	23.563	2.497	0.351	13.390	1.916	0.345
	FODE	16.473	2.142	0.403	19.884	2.377	0.395	21.425	2.490	0.417	12.840	1.947	0.396
	UrbanODE	16.342	2.135	0.406	19.648	2.357	0.394	21.177	2.460	0.408	12.668	1.929	0.391
	UrbanPy	16.082	2.026	0.329	19.025	2.232	0.318	20.810	2.333	0.313	12.336	1.810	0.304
	CUFAR	**14.991**	**1.952**	**0.306**	**18.259**	**2.186**	**0.301**	**19.309**	**2.243**	**0.289**	**11.681**	**1.758**	**0.288**
TaxiNYC	UrbanFM	14.641	1.756	0.338	13.289	1.674	0.325	12.615	1.603	0.334	10.883	1.522	0.326
	DeepLGR	14.512	1.745	0.322	13.686	1.692	0.329	12.709	1.604	0.327	11.063	1.516	0.312
	FODE	14.700	1.777	0.354	13.305	1.695	0.350	12.616	1.639	0.362	11.098	1.556	0.349
	UrbanODE	14.685	1.775	0.352	13.405	1.703	0.350	12.452	1.629	0.357	10.957	1.539	0.335
	UrbanPy	14.323	1.718	0.321	13.484	1.654	0.315	12.268	1.566	0.322	10.910	1.503	0.311
	CUFAR	**13.905**	**1.685**	**0.309**	**13.005**	**1.626**	**0.310**	**12.100**	**1.550**	**0.316**	**10.643**	**1.482**	**0.303**

Results of the five SISR baselines are from Reference [35]. The best results of each dataset are marked in **bold**.

Since SISR baselines performed poorly, we omit them in the remaining experiments. We apply AKR on all FUFI methods (denoted as *continual*), and the results are shown in Table 5. The comparison shows that *continual*-based models consistently outperform *fine-tune*-based models, supporting our motivation that overcoming catastrophic forgetting is vital for the FUFI problem. In addition, *fine-tune* and *continual* both considerably outperform *single-task* (compare the results of Table 4 and Table 5), which suggests that utilizing the knowledge from previous tasks is an effective way to boost the performances of models on new tasks. Interestingly, UrbanFM with *continual* fought back to surpass the UrbanPy on Task 4.

When we incorporate the data augmentation techniques into the adaptive knowledge discriminator, we observe additional performance improvements on all datasets. We speculate this is because the data augmentation reduces the strong dependencies between the coarse- and fine-grained flow maps in the merged sample batch and, as a result, relieves the model from over-fitting on specific urban flow patterns.

5.6.3 Catastrophic Forgetting. To better visualize the forgetting phenomenon on new tasks, we show the training process of our model by using *fine-tune* and *continual* protocols in Figure 4. We have the following findings: For *fine-tune* model, its performances on old tasks are severely degenerated, an obvious consequence of the "catastrophic forgetting." The older the task, the more knowledge it forgets. For *continual* model with our designed AKR algorithm, it successfully alleviates the forgetting problem when learning new tasks. In addition, the *fine-tune* model also faces the overfitting issue on Task-4 at Stage-4 (the validation loss starts to rise), and beyond our expectation, the *continual* model performs surprisingly well. This result suggests another potential benefit of the ARK algorithm.

5.6.4 Backward Preventing Forgetting. To investigate whether our trained model (as well as baselines) on the latest task is forgetting the old knowledge of previous tasks, we use the model trained continually on Task-1→Task-2→Task-3→Task-4 and test its performance on Task-1,

Table 5. Performance Comparison between *fine-tune* and *continual* Protocols on TaxiBJ and TaxiNYC Datasets

		Method	Task-2			Task-3			Task-4		
			MSE	MAE	MAPE	MSE	MAE	MAPE	MSE	MAE	MAPE
TaxiBJ	*fine-tune*	UrbanFM	19.162	2.257	0.322	20.499	2.341	0.325	12.285	1.814	0.314
		DeepLGR	20.571	2.336	0.334	21.845	2.427	0.345	12.820	1.858	0.318
		FODE	19.251	2.323	0.379	20.511	2.410	0.387	12.414	1.895	0.369
		UrbanODE	19.070	2.302	0.371	20.275	2.375	0.372	12.182	1.862	0.361
		UrbanPy	18.822	2.208	0.317	20.117	2.293	0.314	12.088	1.800	0.307
		CUFAR	**17.746**	**2.151**	**0.293**	**18.915**	**2.219**	**0.287**	**11.486**	**1.745**	**0.290**
	continual	UrbanFM	18.477	2.215	0.312	19.809	2.290	0.314	11.919	1.778	0.302
		DeepLGR	19.202	2.292	0.342	19.892	2.331	0.330	11.977	1.819	0.314
		FODE	18.799	2.297	0.370	20.012	2.369	0.373	11.997	1.852	0.359
		UrbanODE	18.735	2.289	0.374	19.779	2.340	0.361	11.924	1.836	0.352
		UrbanPy	18.286	2.193	0.311	19.503	2.264	0.314	11.958	1.787	0.304
		CUFAR	17.616	2.141	0.292	18.840	2.213	0.285	11.420	1.735	0.283
		+DA	**17.562**	**2.137**	**0.289**	**18.785**	**2.211**	**0.282**	**11.388**	**1.731**	**0.280**
TaxiNYC	*fine-tune*	UrbanFM	12.657	1.638	0.320	11.929	1.557	0.325	10.261	1.47	0.309
		DeepLGR	13.011	1.650	0.319	11.801	1.553	0.322	10.627	1.497	0.307
		FODE	12.930	1.673	0.342	11.929	1.585	0.338	10.411	1.503	0.325
		UrbanODE	12.979	1.674	0.339	12.035	1.583	0.342	10.345	1.507	0.330
		UrbanPy	12.934	1.628	0.327	11.607	1.527	0.327	10.219	1.466	0.315
		CUFAR	**12.472**	**1.599**	**0.305**	**11.570**	**1.515**	**0.308**	**10.147**	**1.448**	**0.297**
	continual	UrbanFM	12.387	1.609	0.312	11.367	1.521	0.319	9.899	1.450	0.302
		DeepLGR	12.439	1.617	0.318	11.323	1.518	0.310	10.007	1.455	0.300
		FODE	12.411	1.631	0.330	11.496	1.542	0.326	10.108	1.480	0.316
		UrbanODE	12.397	1.626	0.325	11.371	1.538	0.325	10.073	1.475	0.318
		UrbanPy	12.524	1.605	0.312	11.386	1.516	0.312	9.991	1.449	0.303
		CUFAR	12.153	1.580	0.302	11.199	1.500	0.307	9.959	1.438	0.298
		+DA	**12.072**	**1.577**	**0.300**	**11.053**	**1.491**	**0.305**	**9.805**	**1.431**	**0.294**

All models are initially trained on Task-1. Then, each model is fine-tuned on new tasks. *Continual* indicates our designed AKR is applied. Best results of each dataset and protocol are marked in **bold**.

Task-2, and Test-3. For each model, we, respectively, use *fine-tune* and *continual* protocols for training. Experiments are conducted on TaxiBJ, and the performances of CUFAR and five FUFI baselines are shown in Table 6. We can observe that the models under *fine-tune* protocol face severe forgetting problem on all three tasks. Specifically,

— Compared to five baselines, our proposed CUFAR achieves the lowest inference errors, which demonstrates the effectiveness of our designed spatial-temporal inference network for FUFI.
— *Continual* protocol universally improves the inference performances of all methods by non-trivial margins, which shows that our proposed adaptive knowledge replay training algorithm successfully alleviates the "catastrophic forgetting" on previous tasks and is a general algorithm that improves all methods. In general, CUFAR and baselines are less forgetful on close tasks (e.g., Task-3) and more forgetful on remote tasks (e.g., Task-1), which is in line with our expectation. On Task-1, the MSE of CUFAR with *fine-tune* protocol drops about 22.07% compared to *single-task*. In contrast, the MSE of CUFAR with *continual* protocol is on par with the CUFAR with *single-task*.

Table 6. Results of Catastrophic Forgetting Problem on Previous Tasks

Method	Protocol	Task-1			Task-2			Task-3		
		MSE	MAE	MAPE	MSE	MAE	MAPE	MSE	MAE	MAPE
UrbanFM	*single-task*	16.372	2.066	0.335	19.548	2.284	0.328	21.243	2.398	0.336
	fine-tune	18.800	2.176	0.345	22.239	2.388	0.326	20.667	2.341	0.318
	continual	15.605	2.016	0.335	18.915	2.242	0.326	19.421	2.281	0.311
DeepLGR	*single-task*	17.125	2.103	0.339	21.217	2.386	0.350	23.563	2.497	0.351
	fine-tune	19.382	2.216	0.356	23.613	2.479	0.351	22.075	2.433	0.335
	continual	15.555	2.020	0.339	18.763	2.253	0.338	19.263	2.294	0.336
FODE	*single-task*	16.473	2.142	0.403	19.884	2.377	0.395	21.425	2.490	0.417
	fine-tune	18.907	2.244	0.405	22.340	2.454	0.382	20.864	2.414	0.370
	continual	15.815	2.079	0.382	19.100	2.296	0.365	19.614	2.337	0.358
UrbanODE	*single-task*	16.342	2.135	0.406	19.648	2.357	0.394	21.177	2.460	0.408
	fine-tune	18.949	2.223	0.396	22.292	2.423	0.368	20.596	2.372	0.351
	continual	15.633	2.057	0.374	18.768	2.264	0.352	19.335	2.305	0.343
UrbanPy	*single-task*	16.082	2.026	0.329	19.025	2.232	0.318	20.810	2.333	0.313
	fine-tune	18.683	2.155	0.346	22.127	2.365	0.330	20.411	2.312	0.314
	continual	15.147	1.979	0.325	19.404	2.244	0.319	19.589	2.269	0.310
CUFAR	*single-task*	**14.991**	**1.952**	**0.306**	**18.259**	**2.186**	**0.301**	**19.309**	**2.243**	**0.289**
	fine-tune	**18.300**	**2.121**	**0.327**	**21.262**	**2.312**	**0.308**	**19.371**	**2.245**	**0.288**
	continual	**14.982**	**1.950**	**0.301**	**18.094**	**2.166**	**0.290**	**18.570**	**2.199**	**0.280**

Specifically, we train each model by *fine-tune* protocol or *continual* protocol sequentially on Task-1, Task-2, Task-3, and Task-4. Then, we test the trained model on the previous three tasks, i.e., Task-1, Task-2, and Task-3. If the trained model's performance is inferior than the *single-task* protocol (train independently from scratch on each task as a non-forgetting reference), then we say that a certain degree of forgetting has occurred. Best results are marked in **bold**.

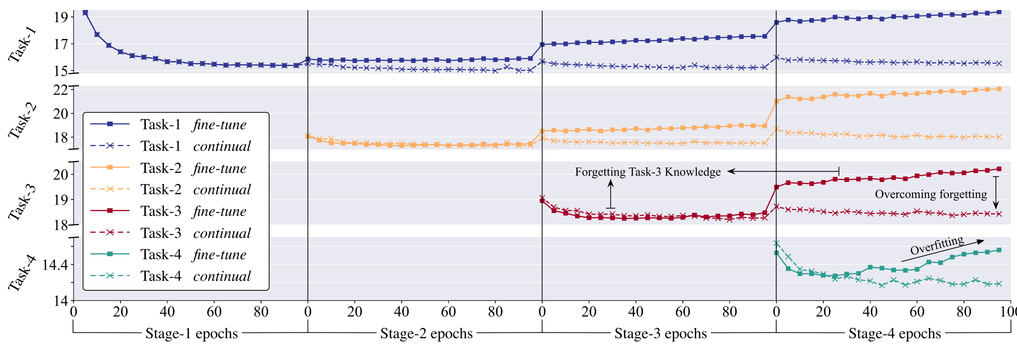

Fig. 4. Visualization of catastrophic forgetting phenomenon. Lines are validation losses (MSE). Each row represents a task. Each column represents a training stage. Each stage, we train the model on current task (by *fine-tune* or *continual* protocol) and validate model on current task and (if any) previous tasks. As we continuously learn on new tasks, the performance of the *fine-tune* model on previous tasks drops severely due to catastrophic forgetting, while *continual* largely alleviates this issue.

— For baselines, the designed AKR training algorithm is more effective for DeepLGR and UrbanODE and is less effective for UrbanPy. For example, UrbanODE with *continual* improves the counterpart (UrbanODE with *fine-tune*) by 19.75%, 20.54%, and 12.74%, in terms of MSE on Task-1, Task-2, and Task-3, respectively.
— We have shown that the knowledge learned from previous tasks can improve the learning process of the new task (cf. Table 5). The results of Table 6 further demonstrate the future knowledge can also improve the performance of the model on old tasks.

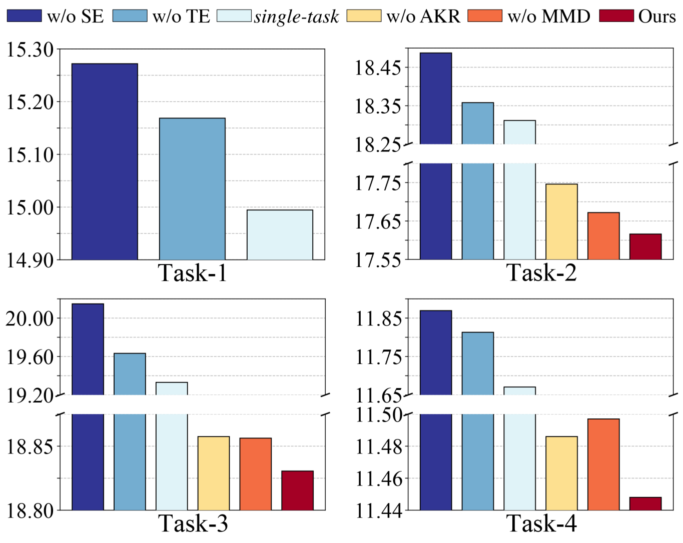

Fig. 5. Ablation study results in terms of MSE.

Combining all the findings above in our context, we would expect CUFAR is able to learn better hidden urban flow map features via simultaneously training on the (1) current flow distributions and (2) previous flow distributions replayed from the memory buffer, resulting in a more robust and generalized FUFI model. Next, we provide additional experimental results to facilitate the understanding of the behaviors of our model, including ablation study, parameter sensitivity, complexity and convergence analyses, and error visualizations.

5.7 Experimental Analysis

5.7.1 Ablation Study. To investigate the contributions of each component in CUFAR, we conduct ablation studies on the following four variants of CUFAR; experiments are conducted on TaxiBJ:

— w/o SE: removing the spatial relation extractor, we evaluate this variant under *single-task* protocol.
— w/o TE: removing the temporal feature extractor, we evaluate this variant under *single-task* protocol.
— w/o AKR: removing the adaptive knowledge replay, i.e., we use *fine-tune* protocol.
— w/o MMD: removing the maximum mean discrepancy, we evaluate this variant under *continual* protocol.

Figure 5 shows the ablation results and we have following remarks: (1) The spatial and temporal feature extractors in CUFAR contribute the most. The combination of the two extractors (i.e., the *single-task* model) is superior than either of them alone, demonstrating the effectiveness of the designed inference network. (2) The results of CUFAR along with CUFAR w/o MMD show that the knowledge learned from previous tasks can significantly improve the FUFI performance. Besides, the designed MMD distance in AKR avoids the "negative replaying" issue and enhances the robustness of the algorithm (as the results of Task-4 show).

5.7.2 Joint Protocol. Training all current and previous tasks simultaneously is often considered to be a powerful approach and serves as a soft upper bound of performance [13]. However, as we show in Figure 6, *joint* training has two main shortcomings: (1) When there are too many tasks, the

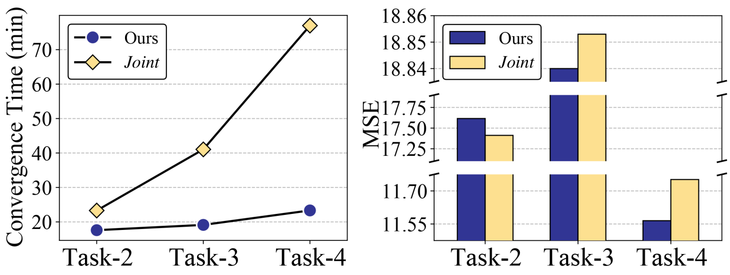

Fig. 6. Comparison between *joint* and *continual* (ours) in terms of convergence time and inference error.

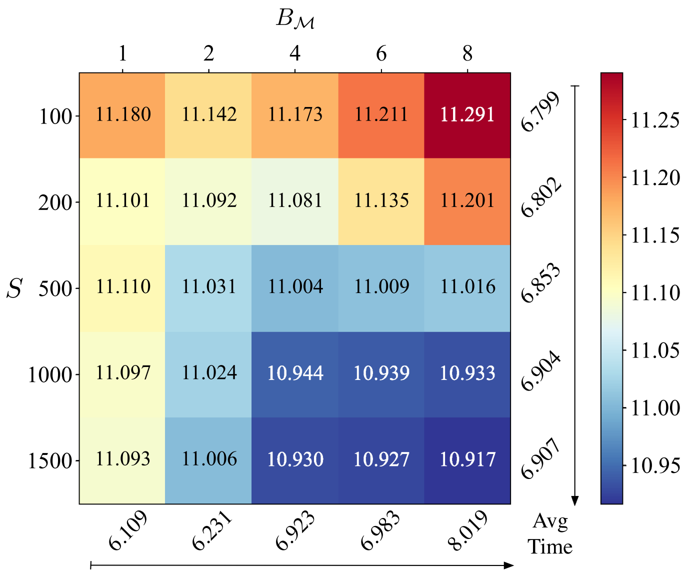

Fig. 7. Sensitivity of buffer mini-batch size $B_{\mathcal{M}}$ and memory buffer size S. MSE results are reported in each grid. The rotated numbers represent average training time (s/epoch) of each column/row. Cell colors indicate model performance.

computation overhead becomes unaffordable, e.g., *joint* spends 3× more training time than CUFAR on four tasks. (2) Noisy data introduced from older tasks may hinder the inference performance if no selection strategy is applied. We can see from the figure that CUFAR equipped with AKR surpasses its *joint* counterpart on Task-3 and Task-4.

5.7.3 *Parameter Sensitivity.* Two crucial hyper-parameters in CUFAR are the buffer mini-batch size $B_{\mathcal{M}}$ and the memory buffer size S, which control the strength of the sample replaying. We vary $B_{\mathcal{M}}$ from $\{1, 2, 4, 6, 8\}$ and S from $\{100, 200, 500, 1,000, 1,500\}$. Figure 7 shows the mean MSE of Task-2 to Task-4 on TaxiNYC, results are the mean of five runs. We have several notable findings: (1) Different settings of S and $B_{\mathcal{M}}$ have a slight influence on training time: The larger the setting values, the more computational consumption of each training epoch. Moreover, the size of $B_{\mathcal{M}}$ has a greater impact on training time than the size of S, because $B_{\mathcal{M}}$ enlarges training batch size. (2) The performance of the model increases as S increases. The improvements are more evident when $B_{\mathcal{M}} = 8$, where we can see that the model performs the worst when $S = 100$ among all combinations but soon becomes the best when $S = 1,500$. This phenomenon demonstrates that the model performs better with a larger $B_{\mathcal{M}}$ and a larger S. (3) The effect of changes in $B_{\mathcal{M}}$ on model

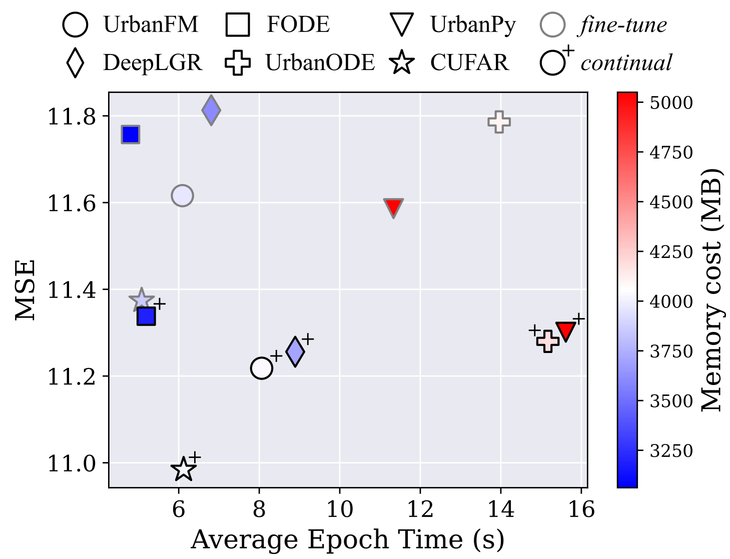

Fig. 8. Complexity comparison of CUFAR and five FUFI baselines. The X-axis and Y-axis represent the average epoch training time and average MSE over Task-2 to Task-4 on TaxiNYC. The color of markers indicates memory usage in the training phase. The + mark indicates we train the model under *continual* protocol rather than *fine-tune*. We can observe that our proposed CUFAR$^+$ model is located in the lower left corner of the figure, achieves the lowest prediction error, and also has low computational time cost.

Table 7. Complexity Comparison, Results Are the Mean MSE of Task-2 to Task-4 on TaxiNYC

Method	Protocol	Memory Cost	Avg Time	MSE
UrbanFM	*fine-tune*	3,965 MB	6.092 s	11.616
	continual	4,040 MB	8.060 s	11.218
DeepLGR	*fine-tune*	3,607 MB	6.808 s	11.813
	continual	3,708 MB	8.893 s	11.256
FODE	*fine-tune*	3,062 MB	4.803 s	11.757
	continual	3,192 MB	5.193 s	11.338
UrbanODE	*fine-tune*	4,102 MB	13.960 s	11.786
	continual	4,180 MB	15.170 s	11.280
UrbanPy	*fine-tune*	6,772 MB	11.335 s	11.586
	continual	6,878 MB	15.614 s	11.301
CUFAR	*fine-tune*	3,858 MB	5.080 s	11.374
	continual	4,018 MB	6.122 s	10.984
	joint	4,430 MB	22.880 s	10.405

Continual indicates our AKR is applied.

performance shows a non-consistent trend and is beyond our expectation. Specifically, a larger $B_{\mathcal{M}}$ decreases the performance when S is small (e.g., 100 or 200) and increases the performance when S is large (e.g., 1,000 or 1,500).

5.7.4 Complexity Analysis. We now empirically analyze the space and time complexities of our model and baselines. In comparison with baselines, AKR algorithm is the new training burden that increases memory and time consumption. The complexity comparison results are shown in Table 7 and Figure 8. Specifically, we have the following observations: (1) Our proposed AKR introduces

Table 8. Results of Forward Knowledge Generalization Learning

Method	Task-2			Task-3			Task-4		
	MSE	MAE	MAPE	MSE	MAE	MAPE	MSE	MAE	MAPE
LwF	17.942	2.166	0.291	19.660	2.262	0.290	11.743	1.767	0.296
EWC	17.812	2.147	0.291	18.977	2.219	0.288	11.536	1.744	0.286
ER	17.742	2.145	0.290	18.861	2.217	0.287	11.485	1.740	0.284
AKR	**17.562**	**2.137**	**0.289**	**18.785**	**2.211**	**0.282**	**11.388**	**1.731**	**0.280**

We sequentially train and validate CUFAR with different continual learning methods on Task-2, Task-3, and Task-4 using the same initial model trained on Task-1. Best results are marked in **bold**.

Table 9. Results of Backward Stability Learning

Method	Task-1			Task-2			Task-3		
	MSE	MAE	MAPE	MSE	MAE	MAPE	MSE	MAE	MAPE
LwF	16.072	2.017	0.316	18.396	2.189	0.297	18.413	2.206	0.286
EWC	18.212	2.109	0.322	21.378	2.310	0.303	19.450	2.243	0.283
ER	15.561	1.988	0.309	18.555	2.194	0.294	18.633	2.208	0.283
AKR	**14.982**	**1.950**	**0.301**	**18.094**	**2.166**	**0.290**	**18.570**	**2.199**	**0.280**

We test the models trained on Task-4 in Table 8 on the previous tasks (Task-1, Task-2, and Task-3). Best results are marked in **bold**.

only a small increase in GPU memory cost about 100 MB for saving samples in the memory buffer $\mathcal{M}$ and sub-memory buffer $\mathcal{M}_{\text{sub}}$. (2) The average training time of each epoch increases about 0.4–4 s when we apply AKR on the model with *fine-tune* protocol. (3) AKR consistently improves performance of each model. The most significant improvements occurred on DeepLGR and Urban-ODE, lowering the MSE about 0.5. (4) As a soft upper-bound method, *Joint* protocol achieves the best performance, however, requires significantly more training time compared to other protocols.

5.7.5 *Comparison to Other Continual Learning Approaches.* To verify the effectiveness of our proposed AKR continual learning algorithm compared to other continual learning methods for the FUFI task, we apply the following representative continual learning methods on CUFAR:

- ***ER (Experience Replay)*** [10]: is a traditional experience replay-based continual learning approach. It maintains a memory of past experience to replay while training on new tasks, thereby reducing knowledge forgetting. Unlike AKR, ER lacks an adaptive knowledge discriminator, which may result in the retention of noisy samples that lead to "negative replaying."
- ***LwF (Learning without Forgetting)*** [34]: is a knowledge distillation-based continual learning method, aiming to acquire new task knowledge without the need for direct access to old data.
- ***EWC (Elastic Weight Consolidation)*** [27]: is a regularization-based continual learning approach that mitigates catastrophic forgetting by constraining changes in important model parameters that learned from previous task(s).

Table 8 and Table 9 present the comparison results of these continual learning approaches in terms of new task forward generalization and old task backward stability, respectively. Our findings demonstrate that the AKR algorithm outperforms the other methods in both forward task learning and backward task retention, indicating superior FUFI inference performance. Specifically, ER exhibits lower inference errors compared to LwF and EWC, supporting our expectation that replay-based methods are more adept at handling the continual FUFI problem. LwF, constrained by its loss function, shows limited generalization when learning new tasks. EWC performs well for the

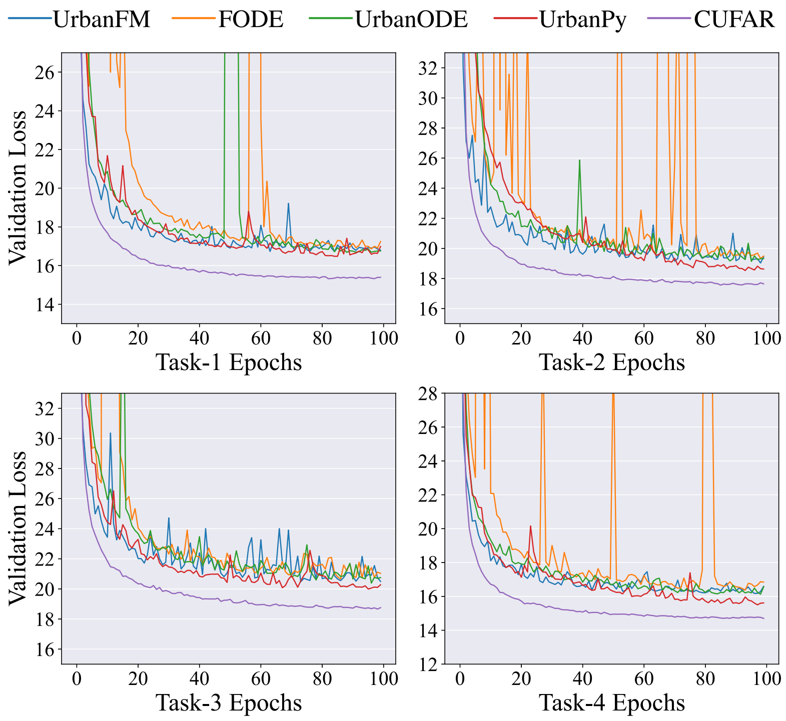

Fig. 9. Convergence speeds of several models on TaxiBJ.

forward learning tasks; however, its backward retention significantly declines, illustrating a pronounced disparity between "stability" (retention of old knowledge) and "elasticity" (adaptability to new knowledge). In summary, the AKR algorithm not only excels in adapting to new knowledge but also effectively mitigates catastrophic forgetting of old knowledge, distinguishing itself from other continual learning approaches on the FUFI problem.

5.7.6 Convergence Analysis. Figure 9 shows the validation loss (MSE) during the training phase of CUFAR and baselines on all tasks. Our model converges smoother and faster than baselines while also having the lowest validation losses. For ODE-based models FODE and UrbanODE, their loss curves oscillated drastically, probably because of the gradient explosion that occurs when solving the ODE functions. UrbanPy, an extension of UrbanFM, employs a cascading strategy that progressively upsamples the coarse-grained flow map, outperforming other baselines. It is worth mentioning that all baselines are less stable on Task-3. This result may be explained by the fact that the flow volumes in TaxiBJ-P3 are larger than that in other tasks. Surprisingly, the loss curve of CUFAR keeps steady and smooth as well as on other tasks.

5.7.7 Error Visualization. Figure 10 shows the inference errors $||\widetilde{\mathbf{X}}_{fg} - \mathbf{X}_{fg}||^2$ of CUFAR and baselines on a case flow map of TaxiBJ under *single-task* protocol; the brighter the pixels, the larger the errors. Specifically, in Figure 10(a), we can observe that the CUFAR has much less bright pixels than baselines. On certain areas (we marked them in the white boxes), the upper one is Zhongguancun district; many companies and universities are located there, and the lower one is a national highway that connected two overpasses. Both two districts have high traffic flow volumes and impact the inference performance greatly. In Figure 10(b), we plot 3D inference error bars of

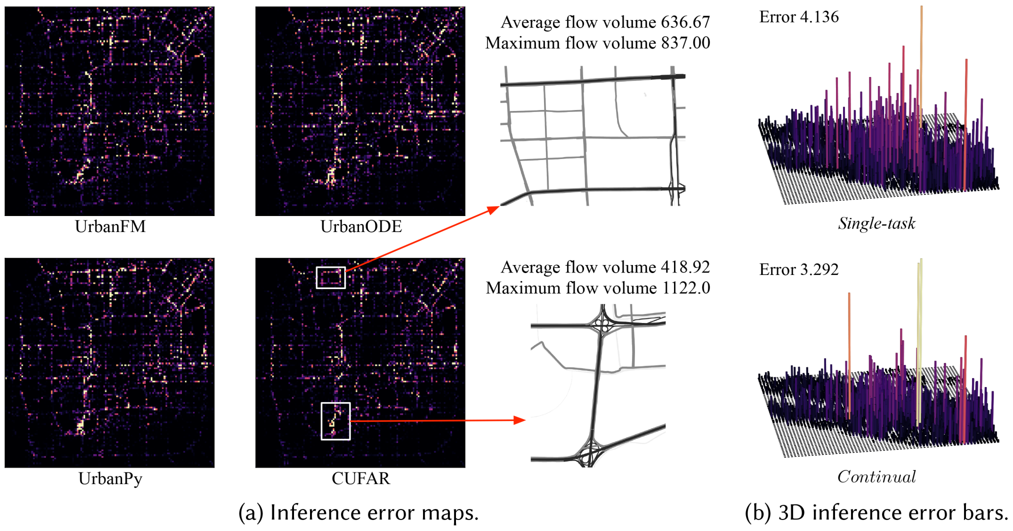

Fig. 10. Visualization of Errors. Fig. 10(a) shows the error maps of different models on TaxiBJ under *single-task* protocol, where brighter pixels indicate larger errors. Fig. 10(b) shows the 3D error bars of CUFAR on TaxiNYC under *single-task* and *continual*, where brighter and higher bars indicate larger errors.

a case map. Compared to *single-task* protocol, *continual* obviously has less higher bars, and its average error is lower than that of *single-task*, which shows that CUFAR's inferred fine-grained flow map is more accurate.

6 CONCLUSIONS

In this work, we presented CUFAR, a novel continual framework for fine-grained urban flow inference. We propose to utilize the learned knowledge from previous tasks to enhance the learning process of the model on new task(s). We designed a spatial-temporal inference network and a general adaptive knowledge replay training algorithm that helps the model alleviate "catastrophic forgetting" and "negative replaying" issues when adapting to new urban flow maps. Extensive experiments on four large-scale real-world FUFI datasets demonstrated the effectiveness and robustness of CUFAR over state-of-the-art baselines. In our future work, we plan to investigate: (1) extending our solution to other urban flow applications/datasets [41]; (2) finding new ways to selectively replay the old samples that are beneficial for the new task with a theoretical guarantee; (3) designing a new knowledge discriminator that considers the external factors (e.g., date and weather) when calculating the flow distribution difference.

REFERENCES

[1] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. 2017. Expert gate: Lifelong learning with a network of experts. In *CVPR*. 3366–3375.
[2] Michael Arbel, Danica J. Sutherland, Mikołaj Bińkowski, and Arthur Gretton. 2018. On gradient regularizers for MMD GANs. In *NeurIPS*.
[3] Mikołaj Bińkowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. 2018. Demystifying MMD GANs. In *ICLR*.
[4] Azzedine Boukerche, Yanjie Tao, and Peng Sun. 2020. Artificial intelligence-based vehicular traffic flow prediction methods for supporting intelligent transportation systems. *Comput. Netw.* 182 (2020), 107484.
[5] Azzedine Boukerche and Jiahao Wang. 2020. Machine learning-based traffic prediction models for intelligent transportation systems. *Comput. Netw.* 181 (2020), 107530.

[6] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. 2020. Dark experience for general continual learning: A strong, simple baseline. In *NeurIPS*. 15920–15930.

[7] Zekun Cai, Renhe Jiang, Xinyu Yang, Zhaonan Wang, Diansheng Guo, Hill Hiroki Kobayashi, Xuan Song, and Ryosuke Shibasaki. 2023. MemDA: Forecasting urban time series with memory-based drift adaptation. In *CIKM*. 193–202.

[8] Chengtai Cao, Fan Zhou, Yurou Dai, and Jianping Wang. 2022. A survey of mix-based data augmentation: Taxonomy, methods, applications, and explainability. *arXiv:2212.10888* (2022).

[9] Gail A. Carpenter and Stephen Grossberg. 1987. ART 2: Self-organization of stable category recognition codes for analog input patterns. *Appl. Opt.* 26, 23 (1987), 4919–4930.

[10] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K. Dokania, Phillip H. S. Torr, and Marc'Aurelio Ranzato. 2019. Continual learning with tiny episodic memories. In *ICML Workshop on Multi-task and Lifelong Reinforcement Learning*.

[11] Guangyong Chen, Pengfei Chen, Yujun Shi, Chang-Yu Hsieh, Benben Liao, and Shengyu Zhang. 2019. Rethinking the usage of batch normalization and dropout in the training of deep neural networks. *arXiv:1905.05928* (2019).

[12] Catherine Cleophas, Caitlin Cottrill, Jan Fabian Ehmke, and Kevin Tierney. 2019. Collaborative urban transportation: Recent advances in theory and practice. *Eur. J. Operat. Res.* 273, 3 (2019), 801–816.

[13] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. 2021. A continual learning survey: Defying forgetting in classification tasks. *IEEE Trans. Pattern Anal. Mach. Intell.* 44, 7 (2021), 3366–3385.

[14] Terrance DeVries and Graham W. Taylor. 2017. Improved regularization of convolutional neural networks with cutout. *arXiv:1708.04552* (2017).

[15] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. 2015. Image super-resolution using deep convolutional networks. *IEEE Trans. Pattern Anal. Mach. Intell.* 38, 2 (2015), 295–307.

[16] Yuntao Du, Jindong Wang, Wenjie Feng, Sinno Pan, Tao Qin, Renjun Xu, and Chongjun Wang. 2021. ADARNN: Adaptive learning and forecasting of time series. In *CIKM*. 402–411.

[17] Zheng Fang, Qingqing Long, Guojie Song, and Kunqing Xie. 2021. Spatial-temporal graph ODE networks for traffic flow forecasting. In *SIGKDD*. 364–373.

[18] Ruicheng Feng, Jinjin Gu, Yu Qiao, and Chao Dong. 2019. Suppressing model overfitting for image super-resolution networks. In *CVPR Workshops*.

[19] Qiang Gao, Goce Trajcevski, Fan Zhou, Kunpeng Zhang, Ting Zhong, and Fengli Zhang. 2018. Trajectory-based social circle inference. In *SIGSPATIAL*. 369–378.

[20] Qiang Gao, Fan Zhou, Kunpeng Zhang, Fengli Zhang, and Goce Trajcevski. 2023. Adversarial human trajectory learning for trip recommendation. *IEEE Trans. Neural Netw. Learn. Syst.* 34, 4 (2023), 1764–1776.

[21] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A kernel two-sample test. *J. Mach. Learn. Res.* 13, 25 (2012), 723–773.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*. 770–778.

[23] Weiwei Jiang and Jiayun Luo. 2022. Graph neural network for traffic forecasting: A survey. *Expert Syst. Applic.* (2022), 117921.

[24] Guangyin Jin, Yuxuan Liang, Yuchen Fang, Zezhi Shao, Jincai Huang, Junbo Zhang, and Yu Zheng. 2023. Spatio-temporal graph neural networks for predictive learning in urban computing: A survey. *IEEE Trans. Knowl. Data Eng.* (2023).

[25] Anirudh Ameya Kashyap, Shravan Raviraj, Ananya Devarakonda, Shamanth R. Nayak K, Santhosh K. V., and Soumya J. Bhat. 2022. Traffic flow prediction models–A review of deep learning techniques. *Cogent Eng.* 9, 1 (2022), 2010510.

[26] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. 2016. Accurate image super-resolution using very deep convolutional networks. In *CVPR*. 1646–1654.

[27] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proc. Nat'l Acad. Sci.* 114, 13 (2017), 3521–3526.

[28] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*. 4681–4690.

[29] Ce Li, Rongpei Hong, Xovee Xu, Goce Trajcevski, and Fan Zhou. 2023. Simplifying temporal heterogeneous network for continuous-time link prediction. In *CIKM*. 1288–1297.

[30] Jiyue Li, Senzhang Wang, Jiaqiang Zhang, Hao Miao, Junbo Zhang, and Philip Yu. 2022. Fine-grained urban flow inference with incomplete data. *IEEE Trans. Knowl. Data Eng.* (2022).

[31] Kehan Li, Jiming Chen, Baosheng Yu, Zhangchong Shen, Chao Li, and Shibo He. 2020. Supreme: Fine-grained radio map reconstruction via spatial-temporal fusion network. In *IPSN*. 1–12.

[32] Ting Li, Junbo Zhang, Kainan Bao, Yuxuan Liang, Yexin Li, and Yu Zheng. 2020. AutoST: Efficient neural architecture search for spatio-temporal prediction. In *SIGKDD*. 794–802.

[33] Wendi Li, Xiao Yang, Weiqing Liu, Yingce Xia, and Jiang Bian. 2022. DGD-DA: Data distribution generation for predictable concept drift adaptation. In *AAAI*, Vol. 36. 4092–4100.

[34] Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.* 40, 12 (2017), 2935–2947.

[35] Yuxuan Liang, Kun Ouyang, Lin Jing, Sijie Ruan, Ye Liu, Junbo Zhang, David S. Rosenblum, and Yu Zheng. 2019. UrbanFM: Inferring fine-grained urban flows. In *SIGKDD*. 3132–3142.

[36] Yuxuan Liang, Kun Ouyang, Junkai Sun, Yiwei Wang, Junbo Zhang, Yu Zheng, David Rosenblum, and Roger Zimmermann. 2021. Fine-grained urban flow prediction. In *WWW*. 1833–1845.

[37] Yuxuan Liang, Kun Ouyang, Yiwei Wang, Xu Liu, Hongyang Chen, Junbo Zhang, Yu Zheng, and Roger Zimmermann. 2022. TrajFormer: Efficient trajectory classification with transformers. In *CIKM*. 1229–1237.

[38] Yuxuan Liang, Kun Ouyang, Yiwei Wang, Ye Liu, Junbo Zhang, Yu Zheng, and David S. Rosenblum. 2020. Revisiting convolutional neural networks for citywide crowd flow analytics. In *ECML-PKDD*. 578–594.

[39] Yuxuan Liang, Kun Ouyang, Yiwei Wang, Zheyi Pan, Yifang Yin, Hongyang Chen, Junbo Zhang, Yu Zheng, David S. Rosenblum, and Roger Zimmermann. 2022. Mixed-order relation-aware recurrent neural networks for spatio-temporal forecasting. *IEEE Trans. Knowl. Data Eng.* (2022).

[40] Feng Liu, Wenkai Xu, Jie Lu, Guangquan Zhang, Arthur Gretton, and Danica J. Sutherland. 2020. Learning deep kernels for non-parametric two-sample tests. In *ICML*. 6316–6326.

[41] Xu Liu, Yutong Xia, Yuxuan Liang, Junfeng Hu, Yiwei Wang, Lei Bai, Chao Huang, Zhenguang Liu, Bryan Hooi, and Roger Zimmermann. 2024. LargeST: A benchmark dataset for large-scale traffic forecasting. In *NeurIPS*.

[42] Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. 2022. Non-stationary transformers: Exploring the stationarity in time series forecasting. In *NeurIPS*, Vol. 35. 9881–9893.

[43] David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *NIPS*. 6470–6479.

[44] Zhisheng Lu, Juncheng Li, Hong Liu, Chaoyan Huang, Linlin Zhang, and Tieyong Zeng. 2022. Transformer for single image super-resolution. In *CVPR*. 457–466.

[45] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. 2022. Online continual learning in image classification: An empirical survey. *Neurocomputing* 469 (2022), 28–51.

[46] Kun Ouyang, Yuxuan Liang, Ye Liu, Zekun Tong, Sijie Ruan, Yu Zheng, and David S. Rosenblum. 2022. Fine-grained urban flow inference. *IEEE Trans. Knowl. Data Eng.* 34, 06 (2022), 2755–2770.

[47] Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. 2009. Domain adaptation via transfer component analysis. In *IJCAI*. 1187–1192.

[48] Haoxuan Qu, Hossein Rahmani, Li Xu, Bryan Williams, and Jun Liu. 2021. Recent advances of continual learning in computer vision: An overview. *arXiv:2109.11369* (2021).

[49] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. 2019. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *ICLR*.

[50] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. 2018. Overcoming catastrophic forgetting with hard attention to the task. In *ICML*. 4548–4557.

[51] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. 2016. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*. 1874–1883.

[52] Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *ICLR*.

[53] Thomas Vandal, Evan Kodra, Sangram Ganguly, Andrew Michaelis, Ramakrishna Nemani, and Auroop R. Ganguly. 2017. DeepSD: Generating high resolution climate change projections through single image super-resolution. In *SIGKDD*. 1663–1672.

[54] Jeffrey S. Vitter. 1985. Random sampling with a reservoir. *ACM Trans. Math. Softw.* 11, 1 (1985), 37–57.

[55] Dongjie Wang, Yanjie Fu, Kunpeng Liu, Fanglan Chen, Pengyang Wang, and Chang-Tien Lu. 2023. Automated urban planning for reimagining city configuration via adversarial learning: Quantification, generation, and evaluation. *ACM Trans. Spatial Algor. Syst.* 9, 1 (2023), 1–24.

[56] Senzhang Wang, Jiannong Cao, Hao Chen, Hao Peng, and Zhiqiu Huang. 2020. SeqST-GAN: Seq2Seq generative adversarial nets for multi-step urban crowd flow prediction. *ACM Trans. Spatial Algor. Syst.* 6, 4 (2020), 1–24.

[57] Zhaonan Wang, Renhe Jiang, Hao Xue, Flora D. Salim, Xuan Song, and Ryosuke Shibasaki. 2022. Event-aware multimodal mobility nowcasting. In *AAAI*, Vol. 36. 4228–4236.

[58] Xovee Xu, Zhiyuan Wang, Qiang Gao, Ting Zhong, Bei Hui, Fan Zhou, and Goce Trajcevski. 2023. Spatial-temporal contrasting for fine-grained urban flow inference. *IEEE Trans. Big Data* 9 (2023), 1711–1725.

[59] Xovee Xu, Yutao Wei, Pengyu Wang, Xucheng Luo, Fan Zhou, and Goce Trajcevski. 2023. Diffusion probabilistic modeling for fine-grained urban traffic flow inference with relaxed structural constraint. In *ICASSP*.

[60] Xovee Xu, Fan Zhou, Kunpeng Zhang, and Siyuan Liu. 2023. CCGL: Contrastive cascade graph learning. *IEEE Trans. Knowl. Data Eng.* 35, 5 (2023), 4539–4554.

[61] Jaejun Yoo, Namhyuk Ahn, and Kyung-Ah Sohn. 2020. Rethinking data augmentation for image super-resolution: A comprehensive analysis and a new strategy. In *CVPR*. 8375–8384.

[62] Haoyang Yu, Xovee Xu, Ting Zhong, and Fan Zhou. 2022. Fine-grained urban flow inference via normalizing flows (student abstract). In *AAAI*. 13101–13102.

[63] Haoyang Yu, Xovee Xu, Ting Zhong, and Fan Zhou. 2023. Overcoming forgetting in fine-grained urban flow inference via adaptive knowledge replay. In *AAAI*. 5393–5401.

[64] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. 2019. CutMix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*. 6023–6032.

[65] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *ICLR*.

[66] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *AAAI*. 1655–1661.

[67] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. 2014. Urban computing: Concepts, methodologies, and applications. *ACM Trans. Intell. Syst. Technol.* 5, 3 (2014), 1–55.

[68] Ting Zhong, Haoyang Yu, Rongfan Li, Xovee Xu, Xucheng Luo, and Fan Zhou. 2022. Probabilistic fine-grained urban flow inference with normalizing flows. In *ICASSP*. 3663–3667.

[69] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. 2020. Random erasing data augmentation. In *AAAI*. 13001–13008.

[70] Fan Zhou, Yurou Dai, Qiang Gao, Pengyu Wang, and Ting Zhong. 2021. Self-supervised human mobility learning for next location prediction and trajectory classification. *Knowl.-based Syst.* 228 (2021), 107214.

[71] Fan Zhou, Xin Jing, Liang Li, and Ting Zhong. 2021. Inferring high-resolutional urban flow with internet of mobile things. In *ICASSP*. 7948–7952.

[72] Fan Zhou, Xin Jing, Xovee Xu, Ting Zhong, Goce Trajcevski, and Jin Wu. 2020. Continual information cascade learning. In *GLOBECOM*. 1–6.

[73] Fan Zhou, Liang Li, Ting Zhong, Goce Trajcevski, Kunpeng Zhang, and Jiahao Wang. 2020. Enhancing urban flow maps via neural ODEs. In *IJCAI*. 1295–1302.

[74] Fan Zhou, Hantao Wu, Goce Trajcevski, Ashfaq Khokhar, and Kunpeng Zhang. 2020. Semi-supervised trajectory understanding with POI attention for end-to-end trip recommendation. *ACM Trans. Spatial Algor. Syst.* 6, 2 (2020), 1–25.

[75] Fan Zhou, Xovee Xu, Goce Trajcevski, and Kunpeng Zhang. 2021. A survey of information cascade analysis: Models, predictions, and recent advances. *Comput. Surv.* 54, 2 (2021), 36 pages.

[76] Xinchi Zhou, Dongzhan Zhou, and Lingbo Liu. 2021. TRUFM: A transformer-guided framework for fine-grained urban flow inference. In *ICONIP*. 262–273.

Received 6 April 2023; revised 21 January 2024; accepted 8 April 2024