Fine-grained Urban Flow Inference with Incomplete Data

Jiyue Li, Senzhang Wang, *Member, IEEE,* Jiaqiang Zhang, Hao Miao, Junbo Zhang, Philip S. Yu, *Fellow, IEEE*

Abstract—Fine-grained urban flow inference, which aims to infer the fine-grained urban flows of a city given the coarse-grained urban flow observations, is critically important to various smart city related applications such as urban planning and public safety. Previous works assume that the urban flow monitoring sensors are evenly distributed in space for data collection and thus the observed urban flows are complete. However, in real-world scenarios, sensors are usually unevenly deployed in space. For example, the traffic cameras are mostly deployed at the crossroads and central areas of a city, but less likely to be deployed in suburb. The data scarcity issue poses great challenges to existing methods for accurately inferring the fine-grained urban flows, because they require all urban flow observations to be available. In this paper, we make the first attempt to infer fine-grained urban flows based on the incomplete coarse-grained urban flow observations, and propose a Multi-Task urban flow Completion and Super-Resolution network (MT-CSR for short) to simultaneously complete the coarse-grained urban flows and infer the fine-grained flows. Specifically, MT-CSR consists of the data completion network (CMPNet for short) and data super-resolution network (SRNet for short). CmpNet is composed of a local spatial information based data completion module LocCmp and an auxiliary information based data completion module AuxCmp to consider both the local geographical and global semantic correlations for urban flow data completion. SRNet is designed to capture the complex associations between fine- and coarse-grained urban flows and upsample the coarse-grained data by stacking the designed super-resolution blocks. To gain an accurate inference, two parts are jointly conducted under a multi-task learning framework, and trained in an end-to-end manner using a two-stage training strategy. Extensive experiments on four large real-world datasets validate the effectiveness and efficiency of our method compared with the state-of-the-art baselines.

Index Terms—Multi-task learning, Urban flow inference, Spatio-temporal data.

1 Introduction

Fine-grained urban flows (e.g., taxi flows, bike flows and human trajectories), which depict detailed human mobility patterns in urban areas, are critically important to many smart city related applications such as urban planning, city renewal and traffic management [1]. To obtain the fine-grained urban flow observations, a large number of sensors are required to be deployed in different areas of a city, which will lead to huge expenditure in terms of daily operation and maintenance [2]. For example, to sense city-wide traffic conditions, a large number of cameras need to be deployed at the intersections of the transportation network. However, in reality, the number of deployed sensors for collecting urban flows is usually very limited due to the high cost. Therefore, how to effectively infer the fine-grained urban flows based on the *sparse* and *coarse-grained* observations by sensors has become an important research issue, attracting rising research attention recently [3, 4].

Traditionally, statistic methods are widely used for fine-grained urban flow inference, such as bilinear, bicubic and nearest interpolation [5]. Xu and Zhu [6] designed a tensor decomposition method for urban flow inference. Chen et al. [7] proposed a tensor co-factorization model to detect fine-grained urban events. However, the performance of these methods is less promising as they only consider the urban flow data, but ignore external features such as weather and holiday. Recently, motivated by the great success of super-resolution algorithms in computer vision [8], researchers tried to regard fine-grained urban flow inference as a spatio-temporal data super-resolution problem [9], since the urban flows of the cell regions of a city can be considered as spatial map images. SRCNN [10] for the first time combined bicubic interpolation method with convolutional neural network for image super-resolution. To address the issue of small receptive fields and slow convergence in SRCNN, Kim et al. [11] proposed VDSR by adopting a deep convolutional network and a residual learning strategy. UrbanFM [12] for the first time employed data super-resolution method for urban flow inference. Different from image super-resolution, UrbanFM contained an external factor fusion network to extract external features (e.g., weather, temperature and holiday) and an inference network. To further improve the performance of UrbanFM at high upscaling rates, UrbanPy [13] employed a pyramid architecture consisting of multiple components,

- *S.Z. Wang is with the School of Computer Science and Engineering, Central South University, Changsha, China, 410083.*
- *J.Y. Li and J.Q. Zhang are with College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China, 211106. J.Y. Li is also with Beijing Jingdong Intelligent City Big Data Research Institute JD Intelligent Cities Research and Jingdong iCity (Beijing) Digits Technology Co., Ltd. JD iCity, JD Technology, Beijing, China.*
- *Hao Miao is with the Department of Computer Science, Aalborg University, Denmark.*
- *J.B. Zhang is with Beijing Jingdong Intelligent City Big Data Research Institute JD Intelligent Cities Research and Jingdong iCity (Beijing) Digits Technology Co., Ltd. JD iCity, JD Technology, Beijing, China.*
- *Philip S. Yu is with the Department of Computer Science, University of Illinois Chicago, Chicago, USA.*

Senzhang Wang is the corresponding author. The paper was partially done when the first author was an intern at JD Intelligent Cities Research under the supervision of the fifth author.

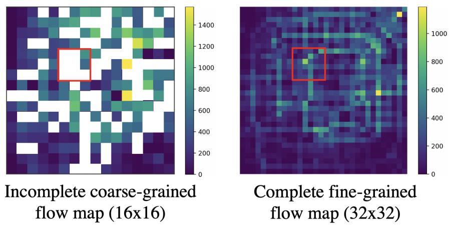


Fig. 1: Fine-grained urban flow inference with incomplete data. The white cell regions denote that the urban flows are unavailable.

where each component functioned as an atomic upsampler for a small scale (e.g. 2×), decomposing the original task into multiple subtasks. Liang et al. [14] designed a general framework named DeepLGR for citywide crowd flow analysis. DeepLGR contained a local feature extraction module, a global context module and a region-specific predictor. However, the major limitation of the above methods is that they all assume that sensors are evenly distributed and the collected coarse-grained urban flows are complete, which may not hold in real application scenarios.

As shown in Fig. 1, this paper makes the first attempt to infer the fine-grained urban flows given that the coarse-grained data are unevenly distributed and incomplete. Compared with the previous urban flow inference problem that ignoring data incompleteness problem, the studied problem is more difficult due to the following challenges.

- **The sparse and incomplete urban flow observations.** Due to the unevenly distributed sensors and data privacy issue, the urban flow observations are sparse and incomplete. It is difficult to accurately infer fine-grained urban flows on the locations without urban flow observations. In addition, urban flow data completion itself is challenging considering the complex spatial correlations of the urban flows in different areas of a city. For example, two areas with similar urban functions (e.g., commercial areas) can be correlated in terms of their urban flow distribution, though they are not spatially adjacent or even far away from each other [15].
- **The complex associations between fine- and coarse-grained urban flow data.** In urban flow maps, one coarse-grained region is associated with multiple (4 in this case) fine-grained neighbor regions, and such association should be considered in fine-grained urban flow inference. Previous work [12] used a structural constraint to guarantee that the flow of a coarse-grained region is equal to the sum of the constituent regions in the fine-grained situation. Nevertheless, this structural constraint may not hold for the studied problem when the urban flow data of some regions are missing. Besides, urban flows can be significantly affected by external factors such as weather and holiday. It is challenging to incorporate both the complex structural constraint and the external features to help the studied problem.
- **Jointly conducting urban flow data completion and super-resolution.** Significantly different from previous works, our work actually contains two tasks: urban flow data completion and super-resolution. Conducting the two tasks separately may not achieve desirable performance as the two tasks are highly correlated. Completing the coarse-grained urban flows can help infer a more accurate fine-grained urban flows, while the inferred fine-grained urban flows can in turn guide us to refine the completion of the initial urban flow observations. Hence, how to jointly conduct both tasks simultaneously rather than separately is also challenging.

To tackle the aforementioned challenges, we present a Multi-Task urban flow Completion and Super-Resolution network model named MT-CSR for the fine-grained urban flow inference with incomplete data. Specifically, we first propose a data completion network (CMPNet) to fill the incomplete coarse-grained urban flow spatial maps by capturing the data correlations among regions. CMPNet includes a local spatial information based data completion module (LocCmp) and an auxiliary information based data completion module (AuxCmp). LocCmp aims to acquire features in the surrounding regions in order to capture the local spatial dependencies. Previous studies [16] show that urban flows are highly correlated with POIs. Thus, we also design AuxCmp to complete the initial urban flow data by capturing the global semantic correlations reflected by POIs. Next, we design a data super-resolution network named SRNet, which contains a feature extraction FE module and stacked super-resolution SR blocks, to capture the complex associations between coarse- and fine-grained urban flows. Concretely, the FE module aims to capture urban flows semantic constraints by extracting external features. Different from previous super-resolution block [12], the interior structure of our SR block is stepped which obtains structural constraints by considering both local and global spatial correlations. Finally, CMPNet and SRNet are jointly conducted under a multi-task learning framework to perform data completion and data super-resolution simultaneously. A two-stage training strategy is also proposed to make the training of MT-CSR more efficient and effective. We summarize our contributions as follow.

- We for the first time study the problem of fine-grained urban flow inference with incomplete data, and propose a Multi-Task urban flow Completion and Super-Resolution network model named MT-CSR to effectively address it.
- We design a data completion network CMPNet to complete the coarse-grained urban flows by considering both the local spatial dependencies and the global POI similarities. We also propose a data super-resolution network SRNet to capture the complex associations between fine- and coarse-grained data.
- Extensive experiments are conducted on four large real-world datasets. Experimental results demonstrate the superiority of our method by comparison with existing state-of-the-art approaches.

The remainder of the paper is organized as follows. We will first briefly review related work in Section 2. Then, notations and problem definition will be introduced in Section 3. Next, we will introduce the proposed model in detail in Section 4, followed by experimental results in Section 5. Finally, we will conclude the paper in Section 6.

2 RELATED WORK

2.1 Urban Flow Data Prediction

Recently, urban flow data prediction [9, 17, 18] has attracted rising research interest in the field of urban computing. Traditionally, statistics-based time series models such as ARIMA and Regression are used to address this problem. For example, [19] used ARIMA model to predict short-term urban flows. However, conventional methods usually are not effective to capture complex spatial and temporal correlations of urban flow data. Recently, various deep learning methods are proposed, and achieve significant prediction performance gains. ConvLSTM [20] employed both spatial and temporal information for spatio-temporal data prediction. It used LSTM [21] as a main model to capture temporal features and then employed CNN on input data and hidden state data for spatial features learning. In order to capture periodic properties, Zhang et al. [22] designed an end-to-end structure named ST-ResNet. ST-ResNet contained a residual neural network and residual convolutional units for spatio-temporal features learning. For learning historical information and making the prediction step by step, Pan et al. [23] designed an encoder-decoder architecture ST-MetaNet+ which combined meta-learning with deep learning. Li et al. [24] proposed an AutoST model for urban flow prediction which consisted of optional convolution operations and learnable skip connections. MT-ASTN [25] adopted a shared-private framework which contained private spatial-temporal encoders, a shared spatial-temporal encoder, and decoders to learn the task-specific features and shared features. Different from urban flow data prediction, the problem studied in this paper aims to infer the fine-grained urban flows based on the coarse-grained observations rather than predicting the future urban flows based on historical observations.

2.2 Spatio-Temporal Data Completion

Spatio-temporal data completion aims to fill the sparse and incomplete spatio-temporal data with reliable values. In the early years, statistic algorithms are used for urban flow data completion [26]. Criminisi et al. [27] proposed a exemplar-based inpainting algorithm to find the nearest patch in the source region and copy the patch of source region to target patch. PatchMatch [28] was an approximate nearest neighbor method which used the consistence of urban flows to reduce the scope of search and self-similar redundancy. He and Sun [29] calculated k main offsets, fused offsets with urban flows and then stacked processed urban flows. However, such methods ignored the spatial-temporal correlations of urban flows. Recently, a large number of deep learning based image completion algorithms are applied to solve urban flow data completion, because urban flows in grid regions of a city can be treated as images [30]. Pathak et al. [31] proposed a context encoder to combine a encoder-decoder structure with reconstruction loss and adversarial loss for generating the contents of arbitrary images conditioned on surroundings. Partial Convolutions (PConv) [32] resolved some deficiencies of existing models like data mismatching and data blurring by using partial convolutions, where the convolution was masked and renormalized to be conditioned on valid data. To address the over smoothness issue, Nazeri et al. [33] proposed EdgeConnect model to gain more ideal images by focusing attention on edges of images and giving hallucinated edges. Nevertheless, the major limitation of the above models is that directly applying them to the studied problem may not achieve desirable performance as urban flows are highly correlated to external features such as weather and holiday.

2.3 Spatio-Temporal Data Super-Resolution

Traditionally, statistic methods are also widely used for spatial-temporal data super-resolution, such as bilinear, bicubic and nearest interpolation [5]. Xu and Zhu [6] designed a tensor decomposition method to achieve spatio-temporal data super-resolution. Chen et al. [7] employed a tensor co-factorization model to make spatio-temporal urban event super-resolution. Recently, approaches [34, 35] for image super-resolution are also applied to spatio-temporal data super-resolution tasks [8], since some spatio-temporal data can be regarded as images. SRCNN [10] was the first end-to-end super-resolution algorithm which combined CNN layers with bicubic interpolation method. To solve the issues of relying on the content of small image areas and the slowness of convergence of SRCNN, Kim et al. [11] proposed VDSR model by adopting a deep convolutional network and a residual learning strategy. Ledig et al. [35] proposed SRGAN, which was the first super-resolution model for $4\times$ upsampling. SRGAN used a perceptive loss to push the inferred super-resolution images to the natural image manifold.

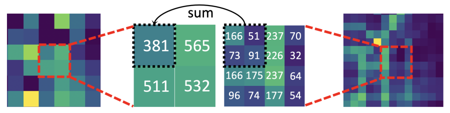
However, image super-resolution is essentially different to spatio-temporal data super-resolution, due to the complex spatio-temporal data correlations and the effect of external features. UrbanFM [12] designed an external factor fusion network to extract external features (e.g. weather and holiday) and it was combined with an inference network. To further improve the performance of UrbanFM at high upscaling rates, Ouyang et al. [13] presented UrbanPy model which employed a pyramid architecture containing multiple components, where each functioned as an atomic upsampler for a small scale like $2\times$. Liang et al. [14] designed a general method named DeepLGR for citywide crowd analytics. DeepLGR designed a local feature extraction module for nearby information, a global context module for broadening range of vision and a region-specific predictor for reducing the number of network parameters. However, existing works on fine-grained urban flow inference all assume that the coarse-grained urban flows are incomplete and thus may not work well when the data is sparse and incomplete.

3 NOTIONS AND PROBLEM DEFINITION

We will first give some notations to help us state the studied problem. Then a formal problem definition will be given.

Definition 1. **Region.** Based on latitude and longitude, we partition a city into a $I \times J$ grid map, and denote all regions as $R = \{r_{1,1}, ...r_{m,n}, ...r_{I,J}\}$ where $r_{m,n}$ is the m-th row and n-th column cell region of the grid map.

Definition 2. **Flow map.** Let $\mathcal{T}$ be a collection of urban flow trajectories. Given a cell region $r_{m,n}$, the corresponding



(a) Coarse-grained flow map (6x6)

(b) Fine-grained flow map (12x12)

Fig. 2: The association between coarse- and fined-grained urban flows.

inflow and outflow map of urban flows in the time slot t are defined as:

$$X^t_{in,m,n} = \sum_{f^t \in \mathcal{T}} \{f^{t-1} \notin r_{m,n} \cap f^t \in r_{m,n}\}$$

$$X^t_{out,m,n} = \sum_{f^t \in \mathcal{T}} \{f^{t} \in r_{m,n} \cap f^{t+1} \notin r_{m,n}\}$$

where $f \in \mathcal{T}$ are urban flows. $f^{t-1} \notin r_{m,n}$ denotes f in time $t-1$ that is not within region $r_{m,n}$ and $f^t \in r_{m,n}$ denotes f in time t is within region $r_{m,n}$. $\cap$ denotes intersection operator. We denote the inflows and outflows of all the regions in t as a urban flow tensor $\mathcal{X}^t \in \mathbb{R}^{2\times I \times J}$.

Definition 3. **Coarse- and fine-grained urban flow spatial map.** A coarse-grained urban flow spatial map represents data granularity we observe based on the original sensors. It is obtained by integrating nearby grids within an $N \times N$ range in a fine-grained urban flow map given an upscaling factor N. Fig. 2 illustrates an example when $N = 2$. Each coarse-grained urban flow in Fig. 2 (a) consists of 2×2 smaller urban flows in Fig. 2 (b). We denote coarse- and fine-grained urban flow maps in time t as $\mathcal{X}^t_{cg} \in \mathbb{R}^{2\times I\times J}$ and $\mathcal{X}^t_{fg} \in \mathbb{R}^{2\times NI\times NJ}$, respectively. Note that coarse-grained spatial maps are incomplete with some entry values missing. We denote the incomplete coarse-grained spatial map as $\mathcal{X}^t_{cg,un} \in \mathbb{R}^{2\times I\times J}$.

Definition 4. **POI feature tensor.** We collect Point of Interest category distribution (e.g., education, residence, and shopping) in each cell region, and denote POI features in all the cell region as a tensor $\mathcal{P} \in \mathbb{R}^{K\times I\times J}$, where K denotes the POI categories.

Based on the above definitions and notations, we formally define the studied problem as follows.

Problem 1. Given an upscaling factor $N \in \mathbb{Z}$, a set of historical incomplete coarse-grained urban flow spatial maps $\{\mathcal{X}^{t-k}_{cg,un}, \mathcal{X}^{t-k+1}_{cg,un}, ..., \mathcal{X}^{t}_{cg,un}\} \in \mathbb{R}^{2\times I\times J}$, POI feature tensor $\mathcal{P}$ and external features matrix $\mathcal{E}$ including weather and holiday, our goal is to infer a complete fine-grained urban flow spatial map $\mathcal{X}^t_{fg} \in \mathbb{R}^{2\times NI\times NJ}$.

The notations and their descriptions used in this paper are listed in Table 1.

4 METHODOLOGY

Fig. 3 shows the framework of MT-CSR, which contains a data completion network CMPNet and a data super-resolution network SRNet. In CMPNet, a local spatial information based data completion module LocCmp and an auxiliary information based data completion module AuxCmp are conducted to capture local spatial and global semantic dependencies, respectively. External features $\mathcal{E}$ including weather and holiday are incorporated into LocCmp module as side information. To capture the global semantic dependencies, we design an AuxCmp module to help fill the incomplete coarse-grained urban flows with the data in the regions with similar POI distributions. We will elaborate this part in Section 4.1.

TABLE 1: Notations and Descriptions

Symbol	Description
R	Regions set
$\mathcal{T}$	Collection of urban flow trajectories
$N \in \mathbb{Z}$	Upscaling factor
$\mathcal{P} \in \mathbb{R}^{K\times I\times J}$	POI features tensor
$\mathcal{E}$	External features matrix
$\mathcal{X}^t_{cg,un} \in \mathbb{R}^{2\times I\times J}$	Incomplete coarse-grained urban flow spatial maps in time slot t
$\mathcal{X}^t_{cg} \in \mathbb{R}^{2\times I\times J}$	Coarse-grained urban flow spatial maps in time slot t
$\mathcal{X}^t_{fg} \in \mathbb{R}^{2\times NI\times NJ}$	Fine-grained urban flow spatial maps in time slot t
$\mathcal{M}^t_{cg,un} \in \mathbb{R}^{2\times I\times J}$	Masks of incomplete coarse-grained urban flow spatial maps in time slot t
$\mathcal{H}^t_{loc} \in \mathbb{R}^{2\times I\times J}$	Output of LocCmp module in time slot t
$s_{r_a,r_b} \in \mathbb{R}$	POI similarity between two regions
$\mathcal{H}^t_{auc} \in \mathbb{R}^{2\times I\times J}$	Output of AuxCmp module in time slot t

Next, the outputs of LocCmp and AuxCmp are fused and input into SRNet. SRNet includes a feature extraction FE module and stacked super-resolution SR blocks, to capture the associations between coarse- and fine-grained urban flows. External features are fed into the Extractor block, aiming to capture the semantic constraints of urban flows. Then $\times 2^m$ with grey dotted square is designed to upscale external features to match data dimension of SR block and we will give details in the following section. The fusion of input and extracted external features is input to stacked SR blocks. In each SR block, Extraction net and UpScale module are designed to extract features and upscale complete coarse-grained urban flows, respectively. Different from previous super-resolution module, we design a stepped structure inside each SR block to capture local and global spatial structure constraints between coarse- and fine-grained urban flows. Finally, we design a distributional upsampling module to guarantee the urban flows in a coarse-grained region being equal to the sum of flows of the constituent regions in the fine-grained situation. This step will be introduced in detail in Section 4.2. CMPNet and SRNet are jointly conducted under a carefully designed multi-task learning framework. The final objective function for multi-task learning model will be described in Section 4.3.

4.1 Data Completion Network

In order to capture both local spatial and global semantic correlations for urban flow data completion, we design two modules, local spatial information based data completion and auxiliary information based data completion.

Local spatial information based data completion As shown in Fig. 4, the module tries to fill the incomplete coarse-grained urban flows by considering the local spatial correlations among regions. To conduct data completion

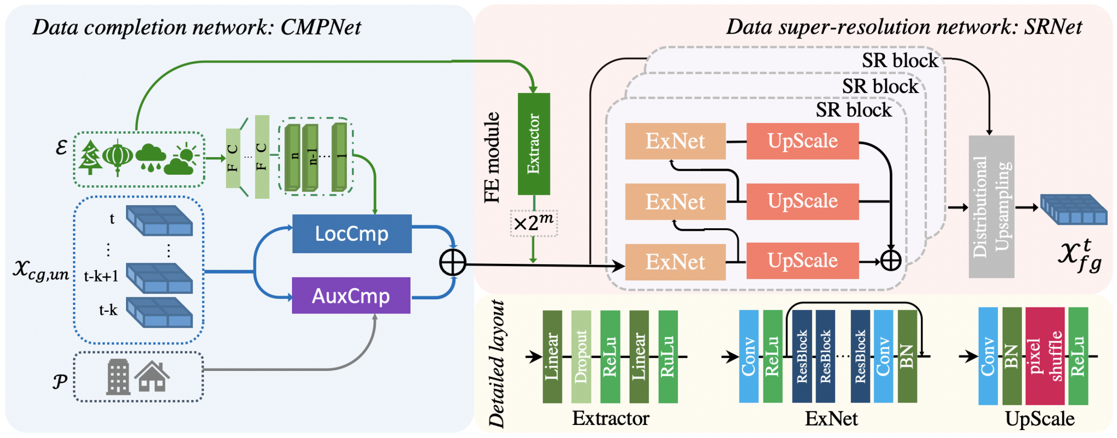


Fig. 3: The MT-CSR model for 2^m upsacling. $\bigoplus$ denotes addition. Note that our model allows arbitrary upsacling factor.

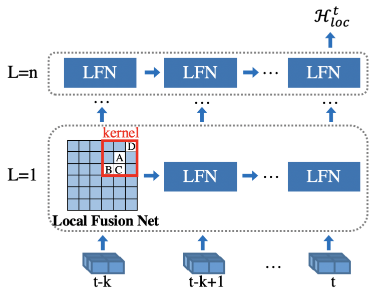


Fig. 4: Illustration of data completion based on local spatial correlations. The regions in white color denote that the data are missing.

over the regions where the data are unavailable, we first define the mask operation as follows:

$$\mathcal{M}^t_{cg,un}(r_{i,j}) = \begin{cases} 1, r_{i,j} = 0 \\ 0, otherwise \end{cases} \quad (1)$$

where $M^t_{cg,un}()$ is a mask function, which marks the regions without crowd flow observations as 1 and the regions with data as 0. Then a local fusion net (LFN) is proposed, whose input is a set of historical incomplete urban flow spatial maps. In LFN, the convolutional kernel extracts features of the receptive field and fills the value of the central region with the convolution result. We take region A whose flow value is unavailable as shown in Fig. 4 as an example to show how to fill its value. A 3×3 kernel is used to conduct convolution operation over the area in the red square shown in the figure whose central region is A, and the convolution result is considered as the estimated value of region A. If there are other cell regions whose values are unavailable (e.g. regions B, C and D in Fig. 4), we set their values as 0 with the mask operation during the convolution operation. Formally, the local spatial information based data completion can be conducted as follows:

$$\mathcal{X}^t_{cg,un,i,j} = \frac{1}{k^2} \sum_{i=i-\lfloor k/2 \rfloor}^{i+\lfloor k/2 \rfloor} \sum_{j=j-\lfloor k/2 \rfloor}^{j+\lfloor k/2 \rfloor} \mathcal{X}^t_{cg,un,i,j} * \mathcal{K} \quad (2)$$

where $\mathcal{X}^t_{cg,un,i,j}$ denotes a region $r_{i,j}$ whose value is missing and needs to be estimated at time t. $\mathcal{K}$ denotes convolutional kernel and k is the kernel size. By setting multiple

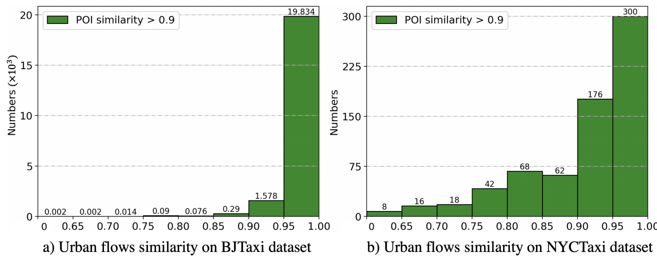


Fig. 5: Distribution of urban flow similarity on BJTaxi and NYCTaxi datasets when POI similarity is larger than 0.9.

layers of LFN vertically, we fill incomplete coarse-grained urban flows based on local spatial correlations. Through stacking LFN horizontally, the temporal dependencies are captured effectively. At the same time, the convolutional kernel with stride setting to 1, moves from left to right and top to bottom. During the kernel moving, the regions with missing values from edge to center are filled one by one. Therefore, the denominator can be set as fixed even if there are missing values in the neighbors of a region. The parameters are shared in the same layer. Inspired by [36, 37] which initializes convolution kernel with external factors, our convolution kernel in LFN is also initialized with external features. As shown in the top left of module in Fig. 3, the external features are fed into multiple fully-connected layers and the output is considered as the convolution filter weights. The formulas are as follows:

$$\mathcal{H}^{t,L} = \sigma(\mathcal{H}^{t-1,L} \odot W_{h,L} + \mathcal{H}^{t,L-1} \odot W_{x,L} + b) \quad (3)$$

$$W_{h,L} = r(f_h(E)) \quad (4)$$

$$W_{x,L} = r(f_x(E)) \quad (5)$$

where L denotes layer, t denotes time, $W_{x,L}$ and $W_{h,L}$ are parameter matrices and $\odot$ denotes convolution operator. When L is equal to 1, $\mathcal{H}^{t,L-1}$ is $\mathcal{X}^t_{cg,un}$. If $t = 1$, we initialize $\mathcal{H}^{t-1,L}$ with random values. $r(\cdot)$ is a reshape function, $f(\cdot)$ is a fully-connected function, $W_{x,L}$ and $W_{h,L}$ are both initialized with external features. The final result of the module is $\mathcal{H}^t_{loc}$.

$$\mathcal{H}^t_{loc} = \mathcal{X}^t_{cg,un} + \mathcal{H}^{t,L} \cdot \mathcal{M}^t_{cg,un} \quad (6)$$

Auxiliary information based data completion We also design an auxiliary information based data completion module to capture the global semantic correlations. Here, we choose to use POIs as the auxiliary features due to the high correlation between POIs and urban flows [16].

We first investigate whether POIs are highly correlated to urban flows and can reflect the trend of urban flows. We conduct similarity analysis between POIs and urban flows over the taxi trip datasets in Beijing and New York city. For convenience, we flatten regions $R = \{r_{1,1}, ...r_{m,n}, ...r_{I,J}\}$ as $\mathrm{R} = \{r_1, ...r_a, ...r_K\}$, where K is equal to $I \times J$.

$$s_{r_a,r_b} = S(r_a, r_b), r_a, r_b \in \mathrm{R} \tag{7}$$

In Equation (7), s_{r_a,r_b} is the POI similarity between regions r_a and r_b. The similarity evaluation metric can be Cosine similarity or Pearson similarity, etc. In this paper we use Cosine similarity as follows.

$$S(r_a, r_b) = \frac{r_a \cdot r_b}{\|r_a\|\|r_b\|}, r_a, r_b \in \mathrm{R} \tag{8}$$

Give an urban flow spatial map $\mathcal{X}_{cg}^t$, for each grid region r_k in the map, we calculate the POI similarity between it and all the other regions. We choose the region pairs whose POI similarity is larger than 0.90, and compute their urban flows similarity to study whether two regions with higher POI similarity also have similar urban flows. Fig. 5 shows the distribution of region pairs in terms of their urban flow similarity when the POI similarity is larger than 0.90. One can see that in Fig. 5a) only 2 pairs of regions whose flow data similarity is smaller than 0.65. The number of region pairs increases with the increase of their urban flow similarity. This figure shows that if two regions have similar POIs, they are highly likely to have similar urban flows. Therefore, we conclude that the POIs are highly correlated to urban flows and reflect the global semantic correlations of regions.

Motivated by the above data analysis, we propose to fill the missing urban flow data based on POI similarity between region pairs. Note that we make all the regions have the same dimension of POI features, with each dimension representing the number of a particular POI type. If there is lacking of POI feature types in one region, we will fill them with zero values. Then we normalize the POI feature values into (0,1). For each region r_k where the urban flow data is missing, we select the region with the largest POI similarity with region r_k as the target region r_*. Next, we compute the similarity between region r_k and its target region r_* based on their urban flow values by Equation (8). Then, we use the most similar urban flows to fill the missing data. Formally, we have

$$s_{r_k,r_*} \geq s_{r_k,r'}, \forall r' \in \mathrm{R}_s, r_k \in \mathrm{R}_t, r_* \in \mathrm{R}_s \tag{9}$$

where r_k is a cell region without urban flow observations, r_* is the region with urban flow observations whose POI similarity is most similar to r_k and r' is an arbitrary region. $\mathrm{R}_t, \mathrm{R}_s$ are the set of regions without and with urban flow data, respectively. Next, we fill incomplete coarse-grained urban flow maps and the completed coarse-grained urban flow map based on AuxCmp is denoted as $\mathcal{H}_{auc}^t$ as follows.

$$\mathcal{H}_{auc}^t = \sum_{k=1}^{|\mathrm{R}_t|} \mathcal{X}_{cg,un}^t(r_k = r_*) \tag{10}$$

where $|\cdot|$ is the size of the set.

LocCmp and AuxCmp capture local spatial and global semantic dependencies, respectively. Next, we employ a linear combination function as follows to integrate them:

$$\mathcal{X}_{cg}^t = W_1\mathcal{H}_{loc}^t + W_2\mathcal{H}_{auc}^t \tag{11}$$

where W_1 and W_2 are weight parameters.

4.2 Urban Flow Data Super-Resolution Network

The urban flow data super-resolution network (SRNet) aims to capture the complex semantic and structural associations between coarse- and fine-grained urban flows for fine-grained urban flows inference. As shown in the middle part of Fig. 3, the output of data completion network is input into SRNet. As shown in the right part of Fig. 3, SRNet first extracts external features by Extractor. Next, $\times 2^m$ with grey dotted square is designed to upscale external features. Then we fuse extracted external features with input data and input it into stacked super-resolution blocks (SR block), where each block functions as a basic upsampler for a small scale (e.g. 2×). Different from previous super-resolution block like linear and pyramid structure, we design a novel stepped structure SR block to capture structural constraints. After multiple SR blocks, we design a distributional upsampling module to make sure the sum of urban flows in a coarse-grained region being equal to the constituent regions in the fine-grained situation. Next, we will introduce the SRNet in details.

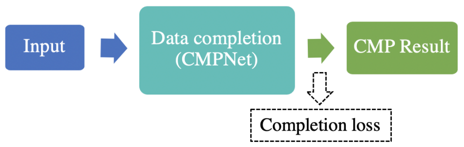
Feature extraction module We use the Extractor to extract the low-level features from the original feature maps. Additionally, combining urban flows with external features will help capture semantic correlations. The Extractor is composed of two linear modules followed by a ReLu function. A Dropout function is between the first linear and ReLu. Then $\times 2^m$ with grey dotted square is used to upscale feature maps to match data dimension of SR block. Finally, extracted features with upscaling operation are fed into each SR block for semantic constraints. Formally,

$$e_m = up_2^{m-1}(Ext(E)) \tag{12}$$

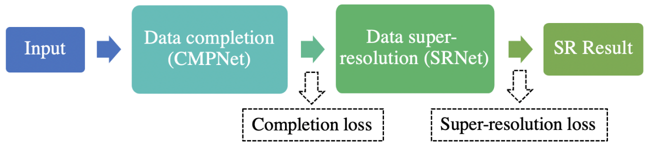
where m is the number of SR block. If $m = 1$, e_1 is fed into the first SR block and if $m = 2$, e_2 is fed into the second SR block. $Ext(\cdot)$ is Extractor, $up_2(\cdot)$ is 2× upscaling operation. The superscript of the function, like m in up_2^{m-1}, is the time for calling function, which is equal to the number of SR blocks.

Super-resolution block Different from previous super-resolution structure, we design a stepped structure inside each SR block. The SR block contains multiple layers. Each layer contains an extraction net ExNet and an upscaling block UpScale to capture spatial associations between coarse- and fine-grained urban flows.

In ExNet, we use a convolution layer followed by a ReLu function to extract low-level features of data. In order to capture spatial correlations among urban flows, we use the ResBlock module proposed in [12] which contains two convolution layers followed by Batch Normalization individually, and ReLu function is in the middle. Then n ResBlock units with the same layout take the low-level feature maps as input and construct high-level feature maps,



(a) Pretraining: data completion network



(b) Training: data completion network and data super-resolution network

Fig. 6: Two-stage training of the proposed MT-CSR: 1) in the first stage, the data completion net is pretrained; 2) in the second stage, the entire network are trained in an end to end manner.

which enlarge the receptive field and capture broader spatial correlations field. Skip connection is used to maintain original information, which helps avoid overfitting. After gaining high-level feature maps, we then use a convolution layer followed by one Batch Normalization. Note that the number of ExNet in each layer is corresponding to the number of layer, for example, the first layer contains one ExNet and the second layer contains two ExNets. More ExNets will provide larger receptive field, and thus the sum of all layers can capture near and far spatial urban flows features simultaneously.

In UpScale, we leverage one convolution layer followed by a Batch Normalization. Next, a pixel shuffle module is used to upsample the urban flow maps by 2× and then a ReLu function is applied to achieve non-linearity. Then we make a weighted summation of each layer in a SR block. The main equations are shown as follow:

$$\begin{aligned} sr(x) = &\lambda_1 up_2^1(ExN^1(x)) \\ &+ \lambda_2 up_2^1(ExN^2(x)) \\ &+ ... + \lambda_l up_2^1(ExN^l(x)) \end{aligned} \tag{13}$$

$$\mathcal{H}_{fg}^t = sr(\mathcal{H}_{m-1}^t + e_{m-1}) \tag{14}$$

where $sr(\cdot)$ is the SR block, $up_2^1(\cdot)$ is the 2× upscaling operation. $ExN(\cdot)$ is the ExNet block and l is the number of layers. λ_1, λ_2, ..., and λ_l are weights of different layers in an SR block. $\mathcal{H}_{m-1}^t$ is the input of the m-th SR block. If $m = 1$, $\mathcal{H}_{m-1}^t$ is $\mathcal{X}_{cg}^t$ from CMPNet.

Distributional upsampling To upsample the fine-grained urban flows, distributional upsampling is usually used [12], which does not provide the exact fine-grained flows but a flow distribution. To better reflect the structural constrains, the flow volume in each fine-grained region is expressed as a fraction of the flow volume in the corresponding coarse-grained region. We also adopt distributional upsampling as follows:

$$\mathcal{H}_{fg,i,j}^t = \mathcal{H}_{fg,i,j}^t / \sum_{\substack{i'=(\lfloor i/N \rfloor -1)*N+1,\\ j'=(\lfloor j/N \rfloor -1)*N+1}}^{\substack{i'=\lfloor i/N \rfloor *N,\\ j'=\lfloor j/N \rfloor *N}} \mathcal{H}_{fg,i',j'}^t \tag{15}$$

$$\mathcal{X}_{fg}^t = up_2^k(\mathcal{X}_{cg}^t) \otimes \mathcal{H}_{fg}^t \tag{16}$$

where $\mathcal{H}_{fg,i,j}^t$ denotes a distributional fine-grained flow map at time t, i, j, i', j' represent region ID and N denotes upsampling factor. $\mathcal{X}_{cg}^t$ and $\mathcal{H}_{fg}^t$ are the outputs of CMPNet and SRNet, respectively and $\otimes$ is the element-wise multiplication operation. $\mathcal{X}_{fg}^t$ is the result of fine-grained urban flow inference.

4.3 Final Objective Function

To jointly conduct urban flow data completion and super-resolution, we design a multi-task learning network with two-stage training strategy shown in Fig. 6. Not considering an end-to-end framework as the gradient optimization process is complex and it cannot make sure the superior performance of CMPNet module. In the first stage, we first pretrain the data completion network separately to obtain complete coarse-grained urban flows. Then in the second stage the two tasks are jointly conducted through fine-tuning.

The first-stage training uses pixel loss as follows to pretrain the data completion network.

$$L_p = \|\mathcal{X}_{cg}^t - \mathcal{Y}_{cg}^t\|_F^2 \tag{17}$$

where $\mathcal{Y}_{cg}^t$ is ground truth of coarse-grained urban flow spatial map at time t.

In the second stage, we use pixel loss to train the entire network. If we treat urban flow data completion and super-resolution as two independent problems, we can design such a loss function:

$$L_t = \|\mathcal{X}_{fg}^t - \mathcal{Y}_{fg}^t\|_F^2 \tag{18}$$

Since urban flow data completion and super-resolution are highly correlated and cannot achieve superior performance if conducting such two tasks separately. Inspired by Cai et al. [38], we conduct a multi-task learning as follows:

$$L_{t,+} = \lambda_1 \|\mathcal{X}_{cg}^t - \mathcal{Y}_{cg}^t\|_F^2 + \lambda_2 \|\mathcal{X}_{fg}^t - \mathcal{Y}_{fg}^t\|_F^2 \tag{19}$$

where λ_1 and λ_2 are hyper-parameters and $\mathcal{Y}_{fg}^t$ is the ground truth of fine-grained urban flow spatial map in the time slot t.

5 EXPERIMENT

5.1 Experiment Setting

5.1.1 Datasets

We use four datasets for evaluation: BJTaxi, NYCTaxi, NYCBike and CDdidi. The statistics of the datasets are given in Table 2. Note that we use incomplete coarse-grained urban flows as input data, thus we artificially and randomly remove some region values as unavailable regions. The details of the datasets are introduced as follows.

- **BJTaxi [22]** This dataset contains the taxi trips in Beijing covering from March 1 to June 30 in 2015. We partition the entire data into non-overlapping training, validation and test sets by a ratio of 7 : 2 : 1.

TABLE 2: Dataset Description

Dataset	**BJTaxi**	**NYCTaxi**	**NYCBike**	**CDdidi**
Latitude	/	$(40.71, 40.765)$	$(40.71, 40.765)$	$(30.655, 30.727)$
Longitude	/	$(-74.01, -73.972)$	$(-74.01, -73.972)$	$(104.043, 104.129)$
Time span	3/1/2015-6/30/2015	1/1/2015-12/31/2015	1/1/2015-12/31/2015	11/1/2016-11/31/2016
Time interval	30 minutes	1 hour	1 hour	1 hour
Coarse-grained shape	16×16 / 8×8	16×16 / 8×8	16×16	16×16
Fine-grained shape	32×32	32×32	32×32	32×32
Upscaling factor	2 / 4	2 / 4	2	2
POI information	✓	✓	✓	✓

- **NYCTaxi**[1] NYCTaxi dataset contains over 160 million taxi trip records in New York from January 1 to December 31 in 2015. Each taxi trip includes pick-up and drop-off dates/times, pick-up and drop-off locations, trip distance, itemized fare, rate type, payment type, and driver-reported passenger counts. For this dataset, we use the first 11 months data for training and validation, and the last month data for testing.
- **NYCBike**[2] This dataset contains more than 9 million bike trips in New York from January 1 to December 31 in 2015. Each bike trip contains the trip duration, start/end timestamps, and station latitude/longitude. For this dataset, we also use the first 11 months data for training and validation, and the last month data for testing.
- **CDdidi**[3] This dataset contains 1 million DiDi taxi trips in Chengdu, China from November 1 to November 31 in 2016. Each trip contains the start-time, end-time, pickup latitude/longitude, and dropoff latitude/longitude. For this dataset, we use the first 22 days data for training, 6 days data for validation, and the remaining data for testing.
- **External features** The used external features includes weather data (e.g. temperature, windspeed, precipitation, e.t.), holiday and POIs.

5.1.2 Evaluation Metrics

We use two metrics Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) defined as follows to evaluate the performance of our MT-CSR model.

$$MAE = \frac{1}{T}\sum_{t=0}^{T}|\mathcal{X}^t - \mathcal{Y}^t| \tag{20}$$

$$RMSE = \sqrt{\frac{1}{T}\sum_{t=0}^{T}(\|\mathcal{X}^t - \mathcal{Y}^t\|_F^2)} \tag{21}$$

where $\mathcal{X}^t$ is the inferred value at time t and $\mathcal{Y}^t$ is the corresponding ground truth.

5.1.3 Baselines

We compare MT-CSR with nine baseline methods including statistics-based methods, image super-resolution methods and recent state-of-the-art fine-grained urban flow inference models. The details of the methods are introduced as follows.

- **Mean partition (Mean)** We evenly distribute the flow volume of a coarse-grained flow region to $n \times n$ fine-grained regions, where n is the upscaling factor. For example, given the flow volume 1 in a coarse-grained region and $n = 2$, all the 4 corresponding fine-grained regions share the same flow volume 0.25.
- **Historical Average (HA)** Similar to mean partition, we utilize the average of historical urban flows and then evenly distribute coarse-grained urban flows into fine-grained ones.
- **SRCNN [10]** SRCNN is an end-to-end image super-resolution algorithm. It employs bicubic interpolation method with CNN layers. SRCNN contains patch extraction and representation, non-linear mapping and reconstruction.
- **VDSR [11]** VDSR employs a deep network with depth up to 20 and a residual network for image data super-resolution.
- **SRResNet [35]** SRResnet is a generative adversarial network containing a designed adversarial loss and a content loss for image super-resolution.
- **DeepLGR [14]** DeepLGR is a state-of-the-art fine-grained urban flow inference model. DeepLGR consists of local feature extraction module, global context module and region-specific predictor.
- **UrbanPy [13]** UrbanPy employs a pyramid architecture containing multiple components. Each component functions as an atomic upsampler for a small scale, which contains an external factor fusion net, an inference network, a proposal net and a correction net. UrbanPy is the most recent state-of-the-art fine-grained urban flow inference model.
- **CmpDL** CmpDL contains two parts: CMPNet module for spatio-temporal data completion and DeepLGR model for spatio-temporal data super-resolution. The training strategy is as the same as our MT-CSR model.
- **CmpUP** CmpUP also has two parts: CMPNet module and UrbanPy model for data completion and super-resolution separately. And the training strategy is also as the same as our MT-CSR model.

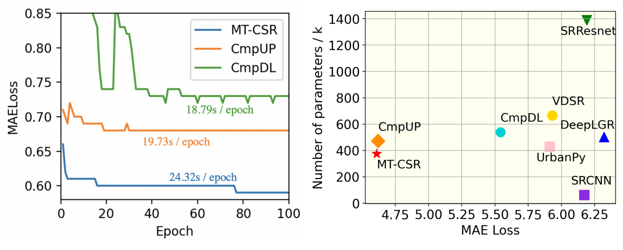
To study whether all the components of our model are useful, we also compare the full version MT-CSR with its variants as follows.

- **CSR** We train urban flow data completion network and super-resolution network separately with Equation (18)

1. https://data.cityofnewyork.us/Transportation/2015-Yellow-Taxi-Trip-Data/ba8s-jw6u
2. https://www.citibikenyc.com/system-data
3. https://outreach.didichuxing.com/app-vue/personal?id=1

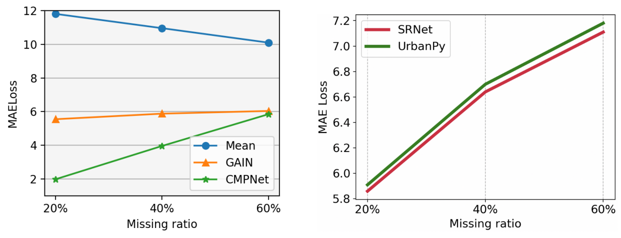
TABLE 3: Comparison results over BJTaxi, NYCTaxi, NYCBike and CDdidi datasets. 2: upsampling factor is 2 from 16×16 to 32×32, 4: upsampling factor is 4 from 8×8 to 32×32. 20%, 40% and 60% denote the rates of the incomplete data. The data are normalized to $(0, 1)$.

Model				Mean	HA	SRCNN	VDSR	SRResNet	DeepLGR	UrbanPy	CmpDL	CmpUP	MT-CSR	Improve
BJTaxi	2	20%	MAE	6.72	6.78	6.17	5.93	6.19	6.32	5.91	5.54	4.62	4.61	0.22%
			RMSE	11.27	11.41	10.34	9.14	10.41	10.47	9.25	9.10	7.69	7.83	-1.82%
			MAPE	34.61%	34.84%	33.32%	33.82%	33.56%	35.33%	34.67%	31.82%	30.66%	28.56%	6.85%
		40%	MAE	7.44	7.48	6.83	6.83	6.93	6.87	6.70	5.44	5.43	5.00	7.92%
			RMSE	12.45	12.53	11.41	11.44	12.12	11.35	10.90	9.50	9.17	8.44	7.96%
			MAPE	37.80%	37.98%	36.55%	36.47%	37.62%	41.42%	37.29%	30.15%	33.40%	29.51%	2.12%
		60%	MAE	7.89	7.92	7.22	7.14	7.29	7.23	7.18	5.79	5.81	5.76	0.52%
			RMSE	12.79	12.85	11.56	11.20	11.81	11.49	11.33	9.83	9.50	9.79	-3.05%
			MAPE	40.95%	41.08%	39.90%	41.64%	39.76%	46.63%	41.29%	32.84%	37.72%	31.88%	2.92%
	4	20%	MAE	6.78	7.04	7.62	76.7	76.6	7.19	5.97	7.07	5.88	5.78	1.70%
			RMSE	11.50	11.90	12.34	12.40	12.07	11.6	9.09	11.33	9.75	9.75	-7.26%
			MAPE	36.20%	36.56%	40.63%	41.52%	42.93%	40.46%	35.46%	38.33%	35.75%	35.45%	0.03%
		40%	MAE	7.56	7.73	7.67	7.66	7.70	6.83	6.74	5.81	5.83	5.80	0.17%
			RMSE	12.67	12.90	12.41	12.42	12.52	11.10	10.69	9.57	9.69	9.76	-1.99%
			MAPE	39.49%	39.79%	41.10%	41.52%	41.21%	41.77%	38.46%	36.18%	36.10%	35.87%	0.64%
		60%	MAE	7.81	7.97	7.91	7.90	7.90	7.14	7.13	5.84	5.84	5.71	2.23%
			RMSE	12.86	13.06	12.68	12.47	12.67	11.12	11.21	9.93	9.33	9.44	-1.18%
			MAPE	41.16%	41.34%	42.70%	43.92%	42.66%	37.77%	40.69%	32.82%	38.31%	37.16%	-13.22%
NYCTaxi	2	20%	MAE	0.61	0.75	0.55	0.56	0.65	0.51	0.36	0.38	0.35	0.33	5.71%
			RMSE	1.18	1.36	1.09	1.11	1.45	0.99	0.81	0.79	0.74	0.65	12.16%
			MAPE	8.84%	10.96%	7.88%	9.50%	8.80%	7.26%	5.40%	5.60%	5.41%	5.55%	-2.78%
		40%	MAE	0.72	0.81	0.64	0.64	0.74	0.62	0.53	0.53	0.56	0.44	16.98%
			RMSE	1.39	1.50	1.26	1.27	6.17	1.26	1.18	1.02	1.08	0.89	12.75%
			MAPE	9.96%	11.41%	8.94%	9.10%	10.24%	8.76%	7.23%	8.09%	8.68%	8.64%	-19.50%
		60%	MAE	0.78	0.85	0.7	0.71	0.75	0.72	0.65	0.67	0.60	0.54	10.00%
			RMSE	1.55	1.61	1.40	1.43	5.56	1.53	1.44	1.27	1.25	1.13	9.60%
			MAPE	10.67%	11.63%	9.61%	12.70%	11.55%	8.84%	10.44%	10.50%	10.42%	10.42%	0.00%
	4	20%	MAE	0.72	0.79	0.60	0.62	0.61	0.42	0.39	0.40	0.42	0.40	-2.56%
			RMSE	1.32	1.43	1.20	1.23	1.25	0.87	0.83	0.80	0.79	0.75	5.06%
			MAPE	10.08%	11.63%	9.17%	9.49%	10.50%	6.20%	5.51%	5.73%	6.01%	5.85%	-6.17%
		40%	MAE	0.77	0.82	0.64	0.66	0.67	0.51	0.51	0.51	0.51	0.5	1.96%
			RMSE	1.47	1.54	1.32	1.35	1.36	1.23	1.24	1.15	1.09	1.03	5.50%
			MAPE	10.45%	11.61%	9.26%	10.84%	9.95%	6.91%	7.16%	7.11%	7.30%	7.51%	-8.68%
		60%	MAE	0.81	0.85	0.70	0.71	0.70	0.52	0.61	0.55	0.54	0.52	3.70%
			RMSE	1.56	1.60	1.42	1.43	1.43	1.24	1.39	1.18	1.10	1.04	5.45%
			MAPE	10.97%	11.79%	9.73%	10.73%	9.85%	7.99%	8.21%	8.04%	8.11%	7.91%	1.62%
NYCBike	2	20%	MAE	0.41	0.49	0.35	0.38	0.35	0.41	0.19	0.23	0.21	0.17	19.05%
			RMSE	1.30	1.49	1.02	1.23	1.04	1.24	0.79	0.85	0.80	0.69	13.75%
			MAPE	5.73%	6.67%	5.07%	6.06%	5.70%	3.46%	3.37%	3.48%	3.08%	3.15%	-2.27%
		40%	MAE	0.42	0.48	0.36	0.35	0.36	0.34	0.24	0.30	0.23	0.22	4.35%
			RMSE	1.39	1.53	1.10	1.09	1.09	1.04	0.95	1.07	0.93	0.89	4.30%
			MAPE	5.54%	6.26%	4.94%	4.93%	5.51%	4.46%	4.79%	4.45%	3.20%	3.17%	0.94%
		60%	MAE	0.44	0.47	0.38	0.38	0.38	0.37	0.30	0.31	0.31	0.28	9.68%
			RMSE	1.55	1.61	1.25	1.27	1.26	1.25	1.19	1.16	1.18	1.12	3.45%
			MAPE	5.36%	5.75%	4.89%	5.08%	4.97%	4.57%	4.64%	4.11%	3.95%	3.61%	8.61%
CDdidi	2	20%	MAE	0.91	1.07	0.81	0.98	0.97	1.07	0.75	1.10	0.94	0.69	8.00%
			RMSE	2.05	2.26	1.79	2.19	2.31	2.28	1.62	1.77	2.13	1.55	4.32%
			MAPE	11.34%	13.28%	12.39%	17.80%	12.64%	12.46%	10.26%	16.75%	12.04%	9.52%	7.21%
		40%	MAE	0.98	1.10	1.02	1.06	1.02	1.11	0.98	0.93	1.09	0.86	7.53%
			RMSE	2.24	2.39	2.31	2.37	2.31	2.19	1.96	2.40	2.37	1.93	1.53%
			MAPE	11.75%	13.47%	12.14%	12.13%	13.97%	13.02%	9.93%	14.89%	13.77%	11.20%	-12.79%
		60%	MAE	1.02	1.11	1.07	1.10	1.07	1.16	1.02	1.24	1.14	0.98	3.92%
			RMSE	2.34	2.46	2.40	2.48	2.41	2.30	2.10	2.00	2.50	2.22	-11.00%
			MAPE	12.02%	13.28%	12.39%	17.21%	12.45%	13.37%	11.40%	13.47%	14.13%	12.25%	-7.46%



(a) Convergence speed (b) Parameter analysis

Fig. 7: Efficiency of MT-CSR model.



(a) CMPNet module analysis (b) SRNet module analysis

Fig. 8: Effectiveness on two tasks of MT-CSR model.

to examine whether the designed multi-task learning framework can improve both tasks.

- **CSR-noAux** We remove the auxiliary information based data completion module from MT-CSR to study whether the POIs are helpful for data completion.
- **CSR-noLoc** We remove the local spatial information based data completion module from MT-CSR to test whether the local spatial information is helpful for data completion.
- **CSR-noPre** We train MT-CSR with the end-to-end strategy by skipping the pretraining stage.

5.1.4 Implementation Details

We implement our model as well as baselines with Pytorch framework on NVIDIA Tesla M40 GPU, and part of the experiment is implemented with Huawei MindSpore framework. We leverage Adam algorithm for gradient descent to perform pretraining and training process with batch size $bz = 16$ and learning rate $lr = 5e-3$. The model parameters are set as follows. In the data completion network, the incomplete coarse-grained urban flow map $\mathcal{X}_{cg,un}^t$ is $5 \times 2 \times I \times J$ for all datasets, where 5 is the previous time slot length, 2 is the number of channels, and $I \times J$ is the size of urban flow maps. The layer of LocCmp is 1, the convolution kernel is 3×3 and we use Cosine similarity in AuxCmp. In the data super-resolution network, the number of ResBlock block is 16 and the layer number inside each SR block is 3.

5.2 Comparison with Baselines

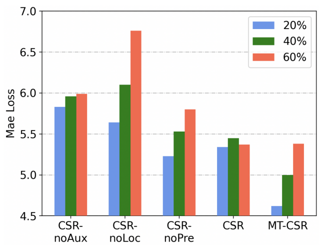
In this section, we compare the model performance against baselines. To more extensively evaluate our model, we set the ratio of incomplete data as 20%, 40% and 60% respectively, and make comparison among different models under each ratio. To illustrate the generalization ability of our model, we set $2\times$ and $4\times$ upscaling factors on BJTaxi and NYCTaxi datasets. As the NYCBike and CDdidi datasets are relatively small and the data are sparse, we do not upscale them with $4\times$.

Performance comparison. The results of different methods are shown in Table 3. The best results achieved among all models are underlined. To more clearly show how much improvement our model achieves, we also present the improvement of MT-CSR comparing to the best results achieved by baselines. It shows that MT-CSR achieves the best result in almost all the cases. On BJTaxi dataset, MT-CSR performs best among all the methods and the improvement is significant. For example, when the upscaling factor is 2 and the missing ratio is 40%, MAE, RMSE and MAPE of MT-CSR are 5.00, 8.44 and 29.51% respectively, improving by 7.92%, 7.96% and 2.21% comparing with the best results achieved by baselines. Mean gives a bad result as it infers urban flows by equal distribution, which ignores both spatial correlations and temporal dependencies. HA achieves the worst performance among all the methods. It is not surprising because different from urban flow prediction which significantly relies on a set of historical urban flows, fine-grained urban flow inference does not rely on historical data much. We can see that SRCNN, VDSR and SRResNet give bad performance comparing with MT-CSR, because they are all designed for image super-resolution, which consider image deblurring problem but ignore spatio-temporal correlations. DeepLGR and UrbanPy are proposed for urban flows inference. However, both of them are ignore data sparsity problem, thus does not perform well. One can see that CmpUP achieves the best performance in terms of RMSE among all the baselines on the BJTaxi dataset when the upscaling factor is 2 and the ratios are 20% and 60%. CmpDL performs best in terms of RMSE when the upscaling factor is 4 and the ratio is 40%. This is because CmpUP and CmpDL use UrbanPy and DeepLGR models respectively, which are especially designed for fine-grained urban flow inference. Similar to the results on the BJTaxi dataset, MT-CSR performs best among all the methods on NYCTaxi, NYCBike and CDdidi datasets. CmpDL and CmpUP perform better than other baselines on the three datasets, because both of them integrate CMPNet module and urban flow inference module.

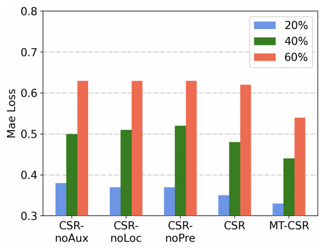
Model efficiency analysis Next, we further discuss the efficiency of our MT-CSR model. We plot the MAE loss curves of different methods in Fig. 7(a) over BJTaxi dataset to compare the computational efficiency of the models. The training time for one epoch of MT-CSR, CmpUP and CmpDL are 24.32, 19.73 and 18.79 seconds, respectively. The result shows that although MT-CSR needs more running time, it converges faster than CmpDL and CmpUP with less number of training epochs, which confirms the efficiency of our two-stage model MT-CSR. We also record the absolute running time for one epoch of our model MT-CSR with $2\times$ and $4\times$ upscaling factors on BJTaxi dataset. The results are 24.32 seconds and 29.66 seconds, respectively. The training time of the model becomes longer when the upsampling scale is larger, because a larger upscaling factor requires we need more SR block in the SRNet module. In order to examine the superiority of MT-CSR, we further compare all the models in terms of their parameter size and the corresponding MAE loss. The result is shown in Fig. 7(b).

TABLE 4: Comparison results on initializing convolutional kernel with external features or random values over BJTaxi dataset.

Missing	20%			40%			60%		
Metric	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Random	2.00	5.64	10.63%	3.99	7.98	12.84%	5.96	9.73	11.30%
Features	1.98	5.58	10.96%	3.96	7.89	12.45%	5.95	9.65	11.22%



a) Results on BJTaxi dataset.



b) Results on NYCTaxi dataset.

Fig. 9: The comparison between MT-CSR and its variants.

TABLE 5: The model performance with different parameter settings over BJTaxi dataset.

		λ_1=1	λ_1=0.1	λ_1=0.01	λ_1=0.001	λ_1=0.0001
λ_2=1	MAE	5.13	4.74	4.62	4.62	4.62
	RMSE	8.65	8.22	7.92	7.86	7.84
λ_2=0.1	MAE	5.49	5.27	4.65	4.62	**4.61**
	RMSE	8.92	8.76	8.06	7.86	**7.83**
λ_2=0.01	MAE	5.29	5.39	6.85	6.56	4.64
	RMSE	8.76	8.90	11.24	10.99	7.91
λ_2=0.001	MAE	5.49	5.42	5.50	8.37	4.71
	RMSE	9.01	8.94	8.98	13.19	8.16
λ_2=0.0001	MAE	5.41	5.45	5.38	6.91	5.26
	RMSE	8.98	8.98	8.95	11.29	8.85

One can see that MT-CSR achieves the lowest MAE 4.61 compared with other models, and at the same time its parameter number is small (only slightly larger than SRCNN). It means that MT-CSR achieves better result with a simpler model (less number of model parameters). SCRNN has the least number of parameters, but its MAE is high.

Model effectiveness analysis on the two tasks. To evaluate the effectiveness of our MT-CSR model, we firstly examine the performance of the data completion module. We compare CMPNet module with mean completion (Mean) and GAIN[39] on BJTaxi dataset. Mean method completes the missing urban flows in a map with the average flows of the entire flow map. The result in Fig. 8(a) shows that CMPNet significantly outperforms the two baselines. The result also shows that with the growth of missing data ratio, less spatio-temporal information is captured by CMPNet, leading to the increase of MAE loss. We also investigate whether SRNet module performs better than state-of-the-art urban flow super-resolution methods. SRNet is an urban flow super-resolution module without data completion. We use Equation (17) to train SRNet. We compare SRNet with UrbanPy on BJTaxi dataset, and the result is shown in Fig. 8(b). The result shows that SRNet consistently and slightly outperforms UrbanPy under different data missing ratios.

5.3 Comparison with Variant Models

In order to examine whether the components in MT-CSR are all beneficial to the studied problem, we compare MT-CSR with its variants including CSR, CSR-noAux, CSR-noLoc and CSR-noPre on BJTaxi and NYCTaix datasets. The result is shown in Fig. 9. One can see that MT-CSR outperforms the four variants. It indicates that the AuxCmp module, the LocCmp module, the pretraining stage and the multi-task learning are all useful, and dropping any one of them will hurt the performance. One can also observe that AuxCmp module contributes most to the model performance improvement. Over BJTaxi dataset, dropping AuxCmp will decrease the performance by up to $20.75\%, 16.11\%, 10.18\%$ in terms of MAE for $20\%, 40\%, 60\%$ missing ratios, respectively. It verifies AuxCmp module is especially important to fine-grained urban flow inference. The inference performance also drops significantly when the LocCmp module is removed, which demonstrates the importance of local information. Moreover, we study the effectiveness of initializing the convolutional kernel with external features in LocCmp module. We compare the external features initialization with random initialization. The model performance with the two kernal initialization methods are given in Table 4. It shows that the performance is slightly improved by initializing the kernal with external features. (reducing MAE from 2.00 to 1.98, from 3.99 to 3.96 and from 5.96 to 5.95 when the missing flow data ratios are 20%, 40%, and 60%, respectively). Training model with an end-to-end strategy by skipping the pretraining stage will hurt the model performance. We can see in Fig. 9 that the CSR-noPre performs worse than the MT-CSR in all missing ratio on both BJTaxi and NYCTaxi dataset, which proves the necessity of pretraining task and our two-stage training strategy. MAE and RMSE of CSR are both much larger than MT-CSR when the incomplete data ratios are 20%, 40% and 60%, which means that the designed multi-task learning framework improves the task. MT-CST combines all these components together and achieves the best performance. Thus one can conclude that the well-designed components in MT-CSR are all useful for the fine-grained urban flow inference problem.

5.4 Parameter Sensitivity Analysis

We next study how sensitive the model is to the deep neural network structure with different parameters. As shown in Table 5, we analyze two hyper-parameters: λ_1 and λ_2 in the final objective function formula (19). We set λ_1 and λ_2 ranging from 1 to 1e-4, respectively. As one can see, with larger λ_1 value, the loss value increases. On the contrary, the loss value decreases with larger λ_2 value. The results shows that SRNet is more important than CMPNet in the training process and we ought to finetune CMPNet slightly. Therefore, in our experiment, we set $\lambda_1 = 1e-4$ and $\lambda_2 = 1e-1$, which achieves a better performance.

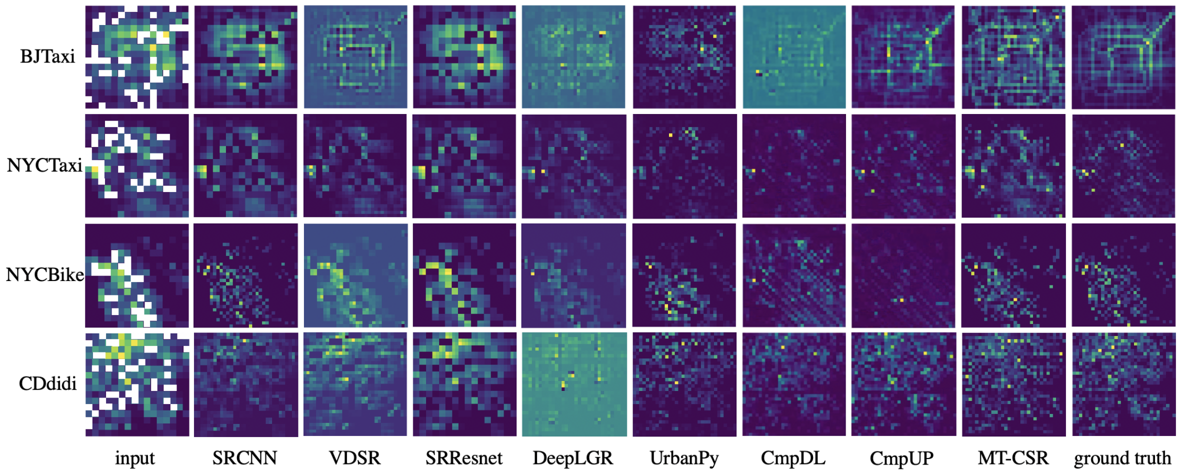
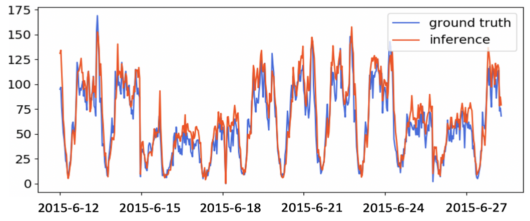
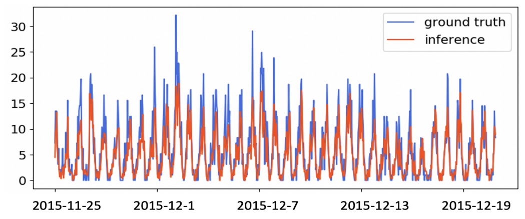


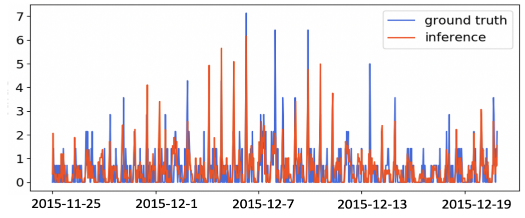
Fig. 10: Visualization of the urban flows inference results with different methods on the four datasets. The input is (16×16) and output is 32×32. The cell regions in white color denote that the urban flow data are unavailable.



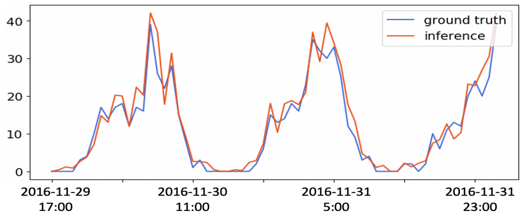
(a) Inflow data on BJTaxi dataset on region $r_{1,8}$.



(b) Inflow data on NYCTaxi dataset on region $r_{0,0}$.



(c) Inflow data on NYCBike dataset on region $r_{6,23}$.



(d) Inflow data on CDdidi dataset on region $r_{0,14}$.

Fig. 11: Inference *vs* ground truth of inflow data on BJTaxi, NYCTaxi, NYCBike and CDdidi datasets.

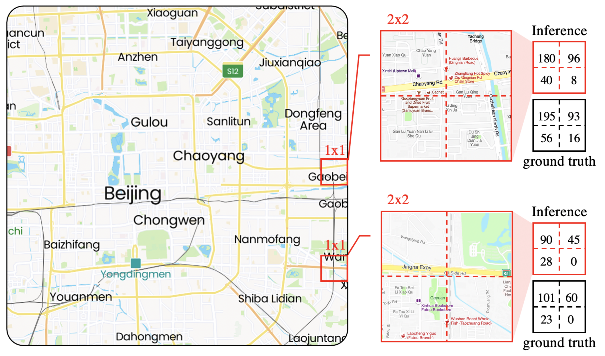


Fig. 12: Visualization of the MT-CSR urban flows inference in Beijing at 2015/6/20 08:30.

5.5 Visualization and Case study

To further intuitively show the model performance, we visualize the inference results with different models and the ground truth. Fig. 10 shows the heat maps of fine-grained urban flow inference results with different models on the four datasets. All input data is 16×16 with 20% missing ratio and the upsampling factor is 2. Taking the results on BJTaxi dataset as an example, SRCNN using bicubic interpolation cannot capture the complex spatial correlations, thus gives a blurry inference. VDSR employing a very deep neural network can easily obtain the general structure of an image. However, its fine-grained urban flow inference is detail-missing. Though SRResNet model enhances VDSR, the designed losses are all useful for content and style of images and even are designed for color distortion. But urban flow observations do not have such characteristics. Thus SRResNet gives an inferior performance on fine-grained urban flow inference. DeepLGR and UrbanPy are superior in inferring fine-grained urban flows when urban flow observations are all available. Thus, DeepLGR and UrbanPy both perform less promising especially for the regions without crowd flow observations. It is obvious that CmpDL and CmpUP perform better than DeepLGR and UrbanPy separately, for both of them consider incomplete data and then add data completion process. However, they still loss some details. Therefore, according to the ground truth of fine-grained urban flow inference, our method MT-CSR gives the superior performance.

To study the scenario that the urban flow data are more likely to be unavailable in suburb regions, next we give a case study by randomly removing some values in the regions of suburbs and test the model performance. We select the suburb regions in Beijing at 8:30 am on June 20 in 2015, and the result is shown in Fig. 12. As one can see, red squares denote suburb regions where the flow data are missing. After data inference with MT-CSR, the fine-grained urban flows are inferred as shown in the right part of the figure. The ground truth is given below the inference result. Compared with the ground truth, the inference result is fairly accurate with small errors. This case study shows that our two-stage MT-CSR model can give accurate fine-grained urban flow inference for the suburb regions where the avaiable flow data are sparser.

To present how close the inferred fine-grained flows are to the ground truth, we visualize the prediction and the ground truth in Fig. 11. From top to bottom, the four figures show the BJTaxi inflow, NYCTaxi inflow, NYCBike inflow and CDdidi inflow, respectively. One can see that the orange curves which are the inferred fine-grained flows can trace the blue curves of the ground truth accurately, including sudden changes, which illustrates that our model MT-CSR can accurately infer the fine-grained urban flow data.

6 Conclusion and Future work

In this paper, we proposed an urban flow completion and super-resolution network for fine-grained urban flow inference with sparse and incomplete data. MT-CSR addressed three challenges which were significant to fine-grained urban flow inference process, including the incomplete urban flows, the complex associations between fine- and coarse-grained data and jointly conducting the two tasks. Extensive evaluations on four real large datasets showed that the proposed model significantly enhanced the performance and outperformed the state-of-the-art models. Studies and visualizations also confirmed the efficiency and effectiveness of our MT-CSR model.

In the future, it would be interesting to explore a more effective urban flow data completion method. Our MT-CSR model is designed for regular regional division of fine-grained flow inference. It is an interesting problem that how to extend the proposed method to irregular regions. A potential solution is that we model the urban flows among irregular regional as graphs, and employ graph neural networks to solve the problem. We will consider it as our future work. We are also interested in further optimizing the model framework. We will consider whether we can design a better deep learning architecture to further improve the performance of fine-grained urban flow inference.

7 Acknowledgement

This work was supported by the National Natural Science Foundation of China (62172443, 62172034), CAAI-Huawei MindSpore Open Fund, and the Beijing Nova Program (Z201100006820053) and was supported in part by NSF under grants III-1763325, III-1909323, III-2106758, and SaTC-1930941.

References

[1] K. Su, J. Li, and H. Fu. Smart city and the applications. In *ICECC*, 2011.

[2] S. Lee, H. Kwon, H. Cho, J. Kim, and D. Lee. International case studies of smart cities: Anyang, republic of korea. *Institutions for Development Sector Fiscal and Municipal Management Division*, 2016.

[3] W. Liu, Y. Yang, E. Wang, and J. Wu. Fine-grained urban prediction via sparse mobile crowdsensing. In *MASS*, 2020.

[4] M. Li, H. Zhang, and J. Chen. Fine-grained dynamic population mapping method based on large-scale sparse mobile phone data. In *MDM*, 2019.

[5] O. Rukundo and H. Cao. Nearest neighbor value interpolation. *International Journal of Advanced Computer Science and Applications*, 2012.

[6] Y. Xu and Y. Zhu. When remote sensing data meet ubiquitous urban data: Fine-grained air quality inference. In *Big Data*, 2016.

[7] L. Chen, J. Jakubowicz, D. Yang, D. Zhang, and G. Pan. Fine-grained urban event detection and characterization based on tensor cofactorization. *IEEE Transactions on Human-Machine Systems*, 2016.

[8] Z. Wang, J. Chen, and S. Hoi. Deep learning for image super-resolution: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[9] S. Wang, J. Cao, and P. Yu. Deep learning for spatio-temporal data mining: A survey. *IEEE transactions on knowledge and data engineering*, 2020.

[10] C. Dong, C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 2015.

[11] J. Kim, J. Kwon Lee, and K. Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *CVPR*, 2016.

[12] Y. Liang, K. Ouyang, L. Jing, S. Ruan, Y. Liu, J. Zhang, D. Rosenblum, and Y. Zheng. Urbanfm: Inferring fine-grained urban flows. In *SIGKDD*, 2019.

[13] K. Ouyang, Y. Liang, Y. Liu, Z. Tong, S. Ruan, D. Rosenblum, and Y. Zheng. Fine-grained urban flow inference. *IEEE transactions on knowledge and data engineering*, 2020.

[14] Y. Liang, K. Ouyang, Y. Wang, Y. Liu, J. Zhang, Y. Zheng, and D. Rosenblum. Revisiting convolutional neural networks for citywide crowd flow analytics. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2020.

[15] X. Zhang, C. Huang, Y. Xu, L. Xia, P. Dai, L. Bo, J. Zhang, and Y. Zheng. Traffic flow forecasting with spatial-temporal graph diffusion network. In *AAAI*, 2020.

[16] L. Wang, X. Geng, X. Ma, F. Liu, and Q. Yang. Cross-city transfer learning for deep spatio-temporal prediction. In *IJCAI*, 2019.

[17] Bowen Du, Hao Peng, Senzhang Wang, Md Zakirul Alam Bhuiyan, Lihong Wang, Qiran Gong, Lin Liu, and Jing Li. Deep irregular convolutional residual lstm for urban traffic passenger flows prediction. *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[18] S. Wang, J. Cao, H. Chen, H. Peng, and Z. Huang. Seqst-

gan: Seq2seq generative adversarial nets for multi-step urban crowd flow prediction. *ACM Transactions on Spatial Algorithms and Systems*, 2020.

[19] S. Lee and D. Fambro. Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting. *Transportation Research Record*, 1999.

[20] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NeurIPS*, 2015.

[21] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[22] J. Zhang, Y. Zheng, and D. Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *AAAI*, 2017.

[23] Z. Pan, W. Zhang, Y. Liang, W. Zhang, Y. Yu, J. Zhang, and Y. Zheng. Spatio-temporal meta learning for urban traffic prediction. *IEEE transactions on knowledge and data engineering*, 2020.

[24] T. Li, J. Zhang, K. Bao, Y. Liang, Y. Li, and Y. Zheng. Autost: Efficient neural architecture search for spatio-temporal prediction. In *SIGKDD*, 2020.

[25] S. Wang, H. Miao, H. Chen, and Z. Huang. Multi-task adversarial spatial-temporal networks for crowd flow prediction. In *CIKM*, 2020.

[26] J. Fan and J. Cheng. Matrix completion by deep matrix factorization. *Neural Networks*, 2018.

[27] A. Criminisi, P. Pérez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on image processing*, 2004.

[28] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 2009.

[29] K. He and J. Sun. Image completion approaches using the statistics of similar patches. *IEEE transactions on pattern analysis and machine intelligence*, 2014.

[30] J. You, X. Ma, D. Ding, M. Kochenderfer, and J. Leskovec. Handling missing data with graph representation learning. *Advances in Neural Information Processing Systems*, 2020.

[31] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.

[32] G. Liu, F. Reda, K. Shih, T. Wang, A. Tao, and B. Catanzaro. Image inpainting for irregular holes using partial convolutions. In *ECCV*, 2018.

[33] K. Nazeri, E. Ng, T. Joseph, F. Qureshi, and M. Ebrahimi. Edgeconnect: Generative image inpainting with adversarial edge learning. In *ICCV*, 2019.

[34] T. Vandal, E. Kodra, S. Ganguly, A. Michaelis, R. Nemani, and A. Ganguly. Deepsd: Generating high resolution climate change projections through single image super-resolution. In *SIGKDD*, 2017.

[35] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017.

[36] B. Yang, G. Bender, Q. Le, and J. Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. In *NeurIPS*, 2019.

[37] D. Kang, D. Dhar, and A. Chan. Incorporating side information by adaptive convolution. In *NeurIPS*, 2017.

[38] J. Cai, H. Han, S. Shan, and X. Chen. Fcsr-gan: Joint face completion and super-resolution via multi-task learning. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 2019.

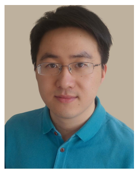
[39] J. Yoon, J. Jordon, and M. Schaar. Gain: Missing data imputation using generative adversarial nets. In *ICML*, 2018.

Jiyue Li received the B.S. degree in Software Engineering from Guangzhou University, Guangzhou, China, in 2020. She is currently a Master student of Nanjing university of aeronautics and astronautics in the department of Computer science and technology. Her research interest includes spatio-temporal data mining and deep learning.

Senzhang Wang received the B.S. and Ph.D. degree in Southeast University, Nanjing, China in 2009 and Beihang University, Beijing, China in 2016 respectively. He is currently a professor at School of Computer Science and Engineering, Central South University. His main research focus is on spatio-temporal data mining, graph data mining and urban computing.

Jiaqiang Zhang received the B.S. degree in Information and Computing Science from Nanjing XiaoZhuang University, Nanjing, China, in 2020. He is currently a Master student of Nanjing university of aeronautics and astronautics in the department of Computer science and technology. His research interest includes spatio-temporal data mining and deep learning.

Hao Miao is a PhD student in the department of computer science at Aalborg University currently, focusing on studying spatio-temporal data analytics. He received the B.S. degree in Computer science and technology from Nanjing Tech university, Nanjing, China, in 2018. His research interest includes spatio-temporal data mining, deep learning and transfer learning.



Junbo Zhang is a Senior Researcher of JD Intelligent Cities Research. He is leading the Urban AI Product Department of JD iCity at JD Technology, as well as AI Lab of JD Intelligent Cities Research. His research interests include Spatio-Temporal Data Mining and AI, Urban Computing, Deep Learning, Federated Learning.

Philip S. Yu received the B.S. Degree in E.E. from National Taiwan University, the M.S. and Ph.D. degrees in E.E. from Stanford University, and the M.B.A. degree from New York University. He is a Distinguished Professor in Computer Science at the University of Illinois Chicago and also holds the Wexler Chair in Information Technology. Before joining UIC, Dr. Yu was with IBM, where he was manager of the Software Tools and Techniques department at the Watson Research Center. His research interest is on big data, including data mining, data stream, database and privacy.