DeepMeshCity: A Deep Learning Model for Urban Grid Prediction

CHI ZHANG, Peking University, Beijing, China
LINHAO CAI, Beijing Juefei Technology Co. Ltd, Beijing, China
MENG CHEN, Peking University, Beijing, China
XIUCHENG LI*, Harbin Institute of Technology, Shenzhen, China
GAO CONG, Nanyang Technological University, Singapore, Singapore

Urban grid prediction can be applied to many classic spatial-temporal prediction tasks such as air quality prediction, crowd density prediction, and traffic flow prediction, which is of great importance to smart city building. In light of its practical values, many methods have been developed for it and have achieved promising results. Despite their successes, two main challenges remain open: a) how to well capture the global dependencies and b) how to effectively model the multi-scale spatial-temporal correlations? To address these two challenges, we propose a novel method—DeepMeshCity, with a carefully-designed Self-Attention Citywide Grid Learner (SA-CGL) block comprising of a self-attention unit and a Citywide Grid Learner (CGL) unit. The self-attention block aims to capture the global spatial dependencies, and the CGL unit is responsible for learning the spatial-temporal correlations. In particular, a multi-scale memory unit is proposed to traverse all stacked SA-CGL blocks along a zigzag path to capture the multi-scale spatial-temporal correlations. In addition, we propose to initialize the single-scale memory units and the multi-scale memory units by using the corresponding ones in the previous fragment stack, so as to speed up the model training. We evaluate the performance of our proposed model by comparing with several state-of-the-art methods on four real-world datasets for two urban grid prediction applications. The experimental results verify the superiority of DeepMeshCity over the existing ones. The code is available at https://github.com/ILoveStudying/DeepMeshCity.

CCS Concepts: • **Information systems → Spatial-temporal systems**.

Additional Key Words and Phrases: Spatial-temporal prediction, crowd/traffic flow prediction, urban computing

1 INTRODUCTION

With the rapid development of sensing technologies, large-scale spatio-temporal data has been produced from mobile devices with GPS (Global Positioning System), traffic sensors, and IoT (Internet of Things) facilities in cities. Urban prediction employs mobility data to achieve accurate traffic forecasting or crowd prediction in a citywide level through cutting-edge deep learning technologies, which is of great importance for traffic management, public safety, and urban planning. The recorded physical values are distributed in any areas accessible to people in a city, not just on the road networks. By meshing a large urban area into numerous fine-grained mesh grids,

*Corresponding author.

Authors' addresses: Chi Zhang, chiizhang@pku.edu.cn, Academy for Advanced Interdisciplinary Studies, Peking University, Beijing, 100871, China; Linhao Cai, cailinhao@juefx.com, Beijing Juefei Technology Co. Ltd, Beijing, 100102, China; Meng Chen, pku.cm@pku.edu.cn, Academy for Advanced Interdisciplinary Studies, Peking University, Beijing, 100871, China; Xiucheng Li, lixiucheng@hit.edu.cn, School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, 518055, China; Gao Cong, gaocong@ntu.edu.sg, School of Computer Science and Engineering, Nanyang Technological University, Singapore, 308232, Singapore.

ACM 1556-4681/2024/3-ART
https://doi.org/10.1145/3652859

we can study and predict a variety of objects, including the density and flow of the crowd or traffic, in a grid-level representation.

Urban grid prediction is a general spatial-temporal forecasting framework in which the urban is partitioned into equally-sized grids and each grid cell represents the physical quantities of interest that change over time. It takes as input multi-step historical grid observations and produces a next-step prediction over all grid cells. Many classic spatial-temporal prediction tasks fall into this setting, e.g., air quality prediction, crowd density and traffic flow prediction, and taxi demand forecasting. The input data can be represented as a 4D tensor $\mathbb{R}^{T\times H\times W\times F}$ in a fashion analogous to video data, where T is the time steps, H, W denotes the index for entire urban grids, and F stands for the physical values of interest such as crowd flow, taxi demand, and traffic accident [12, 14]. In light of its great practical values, many proposals have been developed for it in the past decades. In the early stage, traditional statistical approaches such as Auto-Regressive Integrated Moving Average (ARIMA) [17] and its variants [24] were employed to perform urban prediction. However, these methods study each grid cell individually regardless of the spatial dependencies between different locations. Although the subsequent research considered the spatial-temporal relations [55–57] and external context information (e.g., meteorological data, urban events, and Points-of-Interest (POIs) [28, 35], they often do not well capture the complex non-linear spatial-temporal correlations. To address this, many deep learning-based methods have been proposed and we group them into four categories. (1) Many studies [20, 48–51] treated the citywide mesh grids as a heatmap image, and adopted Convolutional Neural Networks (CNNs) to model the non-linear spatial dependencies owing to the natural Euclidean properties of the grid regions. (2) Some studies [6, 52] used Graph Convolutional Networks (GCNs) to model spatial dependencies by constructing graphs for cities. (3) To further jointly model both spatial and temporal features, several researches [38, 39] combined Recurrent Neural Networks (RNNs) with CNNs. (4) There are also some existing approaches getting better in modeling spatial-temporal correlation with Convolutional Long Short-Term Memory (ConvLSTM) [25] and its variants [10, 12, 13, 46, 59]. Despite their prevalence in the literature, two challenges remain for an accurate urban grid prediction: (a) the existing methods struggle to capture the global spatial dependencies and (b) the spatial-temporal correlations from area scale to grid scale, namely multi-scale spatial-temporal correlations, are not modeled in an effective manner.

The global spatial dependencies mean that a mesh grid cell could be relevant to both neighboring cells and distant cells, due to the convenience of urban transportation such as subways, buses, and taxis. As shown in Figure 1, the city is divided into an equal-sized grid map. G1 contains a subway station and G2 includes work buildings. During working hours, G1 is highly correlated with G2 because people flock to the work buildings from the subway station. In the meantime, since the subway stations in G3 and G1 are on the same metro line (red line on the map), they can be reached directly by commuters. Hence, there is also a strong correlation between G1 and G3 even though they are far away from each other. In previous work [20, 48, 49], CNN-based models were mainly adopted to learn spatial dependencies. However, CNN is more favorable to capture spatial dependencies in neighboring regions due to its inductive bias of the locality [4]. As for the global spatial relationship, although these methods stacked a large number of convolutions or utilized larger kernels and strides to handle it, the effective receptive field is much smaller than the theoretical receptive field [23]. Some attempts [6, 52] tried to construct graphs and leverage GCNs to address the problem. Nevertheless, it is difficult to obtain the accurate spatial topology relationship after meshing the city, which is non-trivial for graph convolution. Therefore, we need a better way to capture the global spatial dependencies effectively.

For the second challenge, there often exist multi-scale spatial-temporal correlations in urban regions. The correlations imply that the distribution of spatial-temporal features at the area scale could provide some prior knowledge (such as urban mobility and human movement) for the distribution at the grid scale in consecutive moments. As shown in Figure 1, the multi-scale regions refer to the grid-scale regions, such as G1 and G2, and area-scale regions, i.e., functional areas composed of spatially contiguous grid cells, such as working area A1 (blue color) and residential area A2 (orange color). If the outflow of people from A1 increases considerably while the

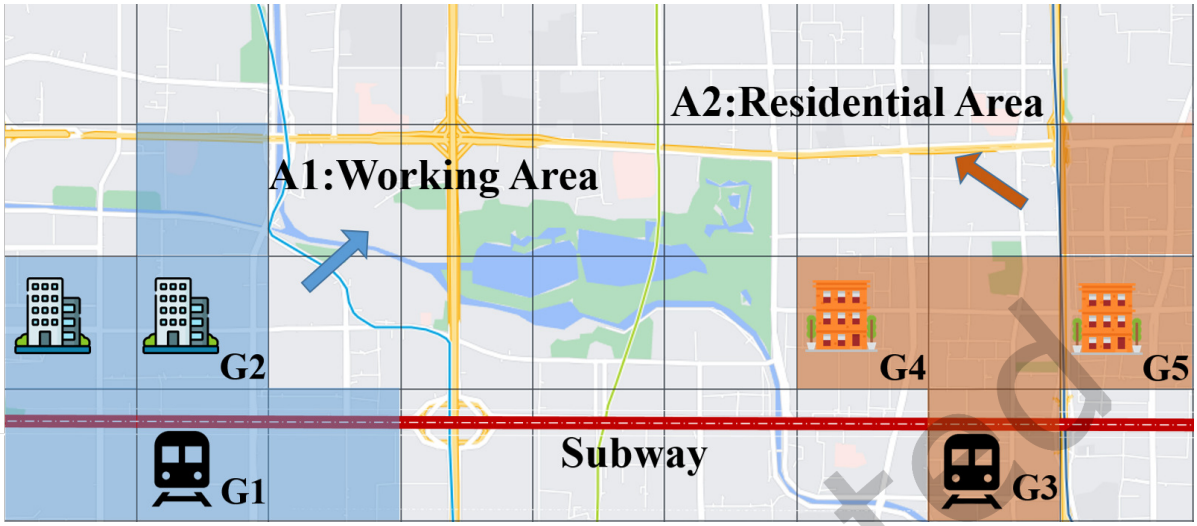

Fig. 1. The example of grid representation and multi-scale correlations. **G1** ∼ **G5** are grid cells, where **G1**, **G3** contains a subway station, **G2** includes work buildings, and **G4**, **G5** covers residential buildings. **A1** and **A2** are working area and residential area respectively, both of which consist of partially similar functional grid cells. The red line across the whole map indicates the subway line, where **G1** and **G3** can be directly accessible to each other.

inflow of crowd from A2 also rises remarkably, it indicates that the current moment is off-time. Then, the crowd density of the residential buildings in G4 and G5 within A2 will demonstrate an upward trend at the next moment. The flow transition from A1 to A2 affects not only the movement of the crowds in the area scale but also the transition pattern in the grid scale (from A2 to G4 and G5). Wang et al. [32] introduced a hierarchical reinforcement learning model that utilizes region-level planning to provide high-level guidance for block-level POI planning in urban land-use planning, yet it does not account for the planning of dynamic flows. Hao et al. [45] proposed a double-branch residual attention network to model the relations in different scales separately regardless of the influence of the area scale on the grid scale. Other proposals [12, 59] employed pyramid ConvLSTMs to capture multi-scale spatial-temporal correlations by exploring the hierarchical features. Unfortunately, some semantic information of area-scale features could be lost after multiple downsampling operations, and could not propagate to grid-scale features effectively by the upsampling with the coarse interpolation. Therefore, the multi-scale spatial-temporal correlations are not captured effectively.

To address the two challenges, we propose a deep learning framework—**DeepMeshCity**—for urban grid prediction. At its core, we design a novel Self-Attention Citywide Grid Learner (SA-CGL) block comprising a self-attention unit and a Citywide Grid Learner (CGL) unit. The self-attention unit aims to capture the global spatial dependencies, and the CGL unit is responsible for learning the spatial-temporal correlations based on the global spatial feature map yielded by the self-attention unit. To effectively capture the multi-scale spatial-temporal correlations, we stack multiple SA-CGL blocks and propose a multi-scale memory unit. Specifically, the multi-scale memory unit travels all SA-CGL blocks in a zigzag path: it flows downwards and across blocks at each time step to learn multi-scale spatial features, and the area-scale features in the top block at the present step are fused with the grid-scale information in the bottom block at the next moment, so as to account for the multi-scale spatial-temporal correlations. Then, we use different single-scale memory units and the multi-scale memory unit of the previous fragment stack as the initialization of the latter in chronological order, to speed up the model

training. Finally, a CNN-based output module is developed to generate the prediction result. In summary, this paper makes the following contributions:

- We propose a general deep learning framework named DeepMeshCity for urban grid prediction, which can capture the global spatial dependencies and the multi-scale spatial-temporal correlations in an effective fashion.
- We design a Self-Attention Citywide Grid Learner (SA-CGL) block which adopts a self-attention unit to model the global spatial dependencies and employs a Citywide Grid Learner (CGL) unit to learn the spatial-temporal correlations.
- We propose a multi-scale memory unit that travels stacked SA-CGL blocks in a zigzag path to represent the multi-scale spatial-temporal correlations.
- Extensive experiments conducted on four real-world datasets verify the efficacy of our proposed method for two urban grid prediction applications, and DeepMeshCity establishes new state-of-the-art results on the four datasets.

2 RELATED WORK

Based on how spatial data is handled as input, the urban prediction could be classified as urban graph prediction and urban grid prediction. The former represents the spatial region by a graph whereas the latter organizes the data by meshing the city into regularly sized grid cells.

2.1 Urban Graph Prediction

Urban graph prediction takes the traffic road network as a graph and the sensors distributed in the roads as the graph signals. Notably, it adopts an external topology graph of sensors as input to represent the connectivity of the whole road network. In this task, GCN-based models [15, 18] are widely adopted to capture both the spatial dependencies and temporal dynamics of traffic data.

GCNs are often combined with other deep learning models in existing research for urban graph prediction. In STGCN [43], it uses GCNs to capture spatial features and TCN [44] with a gated mechanism to learn temporal dependencies. To improve STGCN from the temporal axis, T-GCN [53] substitutes TCN with Gate Recurrent Unit (GRU). Later, ASTGCN [7] and AM-RGCN [47] augment the input length of temporal feature for capturing the periodicity and periodic temporal shift. As for the improvement of spatial dependencies learning, the attention mechanism is considered in model GMAN [54] and MRA-BGCN [3]. Furthermore, Graph WaveNet [36] and AGCRN [1] both adopt an adaptive graph rather than adopting a static one to model the spatial correlations. Other methods such as STSGNN [26] and STFGNN [16] try to model spatial-temporal correlations by constructing the spatial-temporal graph. Recent research applied Transformer to learn the temporal periodicity [2] and long-range dependencies [37] and capture the dynamic spatial dependencies [27] and semantic features [9].

Despite their promising results, these models are only applicable when spatial topology relationships are available, greatly limiting their applications beyond the traffic domain. In contrast, the framework of urban grid prediction is more general and covers various spatial-temporal prediction tasks in urban views.

2.2 Urban Grid Prediction

Urban grid prediction is a general spatial-temporal forecasting problem that covers many prediction tasks such as traffic prediction [6, 40, 52, 58], crowd prediction [12, 48, 59], abnormal event prediction [8, 10, 13], and environment (e.g. air quality and crop yield) prediction [19, 41, 42, 56, 57]. It has been extensively studied in smart cities and is of great importance to both public safety and city management.

Traditional approaches. The traditional statistical approaches such as ARIMA and its variants can be employed to predict taxi-passenger demand [24] and urban human mobility [17]. However, these methods study each

grid cell independently without considering the spatial dependencies between different locations. Zheng et al. [56] proposed a semi-supervised learning approach that adopts an Artificial Neural Network (ANN) to model spatial features and a linear-chain Conditional Random Field (CRF) to represent the temporal dependencies for grid-based air quality prediction. Zhou et al. [58] developed a spatio-temporal Kernel Density Estimation (stKDE) to provide spatial density predictions for ambulance demand. The method employs Intrinsic Gaussian Markov Random Field (IGMRF) to learn the change of the seasonal pattern over time and applies a residual Bayesian network to capture the transition probability among different regions. Moreover, other works [28, 35] used extra context information such as meteorological data, urban events, and POIs to improve the performance of taxi prediction. Although these approaches model the spatial-temporal relations explicitly, they may not well capture the complex non-linear spatial-temporal correlations.

Spatial dependencies. Many studies proposed to treat the citywide mesh grids as a heatmap and adopted CNNs to model the non-linear spatial dependencies by exploring the natural Euclidean properties of the grid regions. Deep-ST [49] firstly applies a convolutional network to capture spatial relations. ST-ResNet [48] is the most representative grid-based method that designs a residual convolutional framework for crowd flow prediction. Based on it, DST-ICRL [5] introduces an irregular convolutional network based on ST-ResNet to capture spatial features, achieving a more accurate urban traffic passenger flow prediction. Then, in order to capture long-range spatial dependencies, DeepSTN+ [20] takes the influence of location function into account with the use of POIs as auxiliary information, and designs a ResPlus block for this purpose. Comparatively, MDL [51] leverages a multitask framework to predict the flows at nodes and on edges collectively and simultaneously. These methods have to rely on stacked layers and larger kernel sizes to capture the global spatial dependencies since the convolution operation is only capable of exploiting the local correlation. However, as the effective receptive fields of CNNs are often much smaller than the theoretical values [23], the aforementioned methods still struggle in modeling the long-range spatial correlations in practice.

Another line of studies applied GCNs to model the spatial dependencies by constructing graphs for cities. Xu et al. [6] proposed a spatiotemporal Multi-Graph Convolution Network (ST-MGCN) for ride-hailing demand forecasting. It adopts multi-graph convolution to model the global spatial correlations, introducing three types of correlations among different regions with graphs, including the neighborhood graph, functional similarity graph, and the transportation connectivity graph. Zhang et al. [52] came up with the Spatial-Temporal Graph Diffusion Network (ST-GDN) for traffic flow prediction. This solution defines a region graph and develops a graph attention network [30] to capture both local and global traffic dependencies between two regions. Nevertheless, it is difficult to obtain the accurate spatial topology relationship in many scenarios after meshing the city, which might severely hinder the performance of graph convolution neural networks.

Spatial-temporal correlations. Prior arts mainly combined CNN-based models with RNN-based models to capture the spatial-temporal correlations. Deep Multi-View Spatial-Temporal Network (DMVST-Net) [39] is a typical framework for taxi demand prediction based on CNN and LSTM. The method applies the local CNN which takes one grid cell and its surrounding cells as input and constructs a weighted graph to capture citywide spatial dependencies. Then, it employs LSTM to capture temporal dependencies from recent time intervals. On the basis of DMVST-Net, Spatial-Temporal Dynamic Network (STDN) [38] designs a flow gating mechanism to explicitly model the dynamic spatial similarity. Besides, it introduces a periodically shifted attention mechanism to address the periodic temporal shift problem. However, these methods could not learn the spatial-temporal correlations effectively by separately modeling the spatial and temporal features.

Other existing works utilized ConvLSTM [25] and its variants [59] to represent the spatial-temporal correlations. Hetero-ConvLSTM [46] is the first work to address spatial heterogeneity based on ConvLSTM for traffic accident prediction. Jiang et al. [10, 13] proposed a Multitask ConvLSTM Encoder-Decoder to model the spatial-temporal correlations from urban human mobility for predicting citywide big events. Recent studies pay more attention to the multi-scale spatial-temporal correlations. Yuan et al. [45] proposed a Multi-View Residual Attention

Network (MV-RANet) which uses a double-branch residual attention network to model the relations in different scales separately regardless of the interaction from the area scale to the grid scale. DeepCrowd [12] employs pyramid ConvLSTMs to capture multi-scale spatial-temporal correlations from pyramid and hierarchical features. Unfortunately, some semantic information of area-scale features may be lost after multiple downsampling, and may not be propagated to grid-scale features efficiently by the upsampling with the coarse interpolation.

3 PRELIMINARY

We present two definitions and introduce two typical prediction tasks under this general urban grid prediction framework.

Definition 3.1 (Urban Grid). We partition the entire urban region into $H \times W$ equally-sized grid cells and denote the observations at a specific time slice t by $X_t \in \mathbb{R}^{H \times W \times F}$ where F indicates the number of physical quantities of interest. We use $x_t^{i,j} \in \mathbb{R}^F$ to represent the observations of grid cell at the i^{th} row and the j^{th} column of X_t, where $0 \leq i \leq H-1, 0 \leq j \leq W-1$.

Definition 3.2 (Urban Grid Prediction). Given the historical observation sequence $[X_{t-\alpha+1}, X_{t-\alpha+2}, \ldots, X_t]$ over the past α consecutive steps and external metadata V^E, urban grid prediction aims to make the prediction of X_{t+1} at the $t+1$ step.

The input feature dimension F can vary against different prediction tasks. In this paper, we instantiate the urban grid prediction with two specific tasks, namely, crowd density prediction and traffic/crowd flow prediction. In the task of crowd density prediction, F equals 1 as each grid cell only records the crowd density; whereas in traffic/crowd flow prediction, F is 2 which corresponds to the inflow and outflow of the grid cell.

4 METHODOLOGY

In this section, we present the details of our proposed method—DeepMeshCity. The overall architecture of DeepMeshCity is depicted in Figure 2. The proposed model has at its core a stack of SA-CGL (Self-Attention Citywide Grid Learner) blocks, which are designed to better handle the global spatial dependencies and the multi-scale spatial-temporal correlations. First, we briefly discuss how to fuse the historical observations with the external metadata that is informative to the evolution of urban observations in Section 4.1. Next, we present the design details of the key module—SA-CGL—with an emphasis on its ability in capturing the global spatial dependencies in Section 4.2. Finally, Section 4.3 illustrates how to model the multi-scale spatial-temporal correlations by stacking a series of SA-CGL blocks.

4.1 Early Fusion for External Metadata

By following the strategy of previous proposals [12, 48], we divide the historical observation sequence into three fragments, namely, $X^H \in \mathbb{R}^{T_h \times H \times W \times F}$, $X^D \in \mathbb{R}^{T_d \times H \times W \times F}$, and $X^W \in \mathbb{R}^{T_w \times H \times W \times F}$, which correspond to the recent hour observations, past day observations, and recent week observations, respectively. T_h, T_d, and T_w are their lengths, whose selection will be explained in Section 5.4.

Apart from the historical observations, there is also external metadata V^E that has a direct impact on urban mobility, such as the date, weather condition, and holiday. We take them as auxiliary input and fuse them with historical observations. This is achieved by feeding V^E into a two-layered MLP with ReLU as the activation function, whose output is denoted by X^E. More formally,

$$X^E = \text{MLP}(V^E). \tag{1}$$

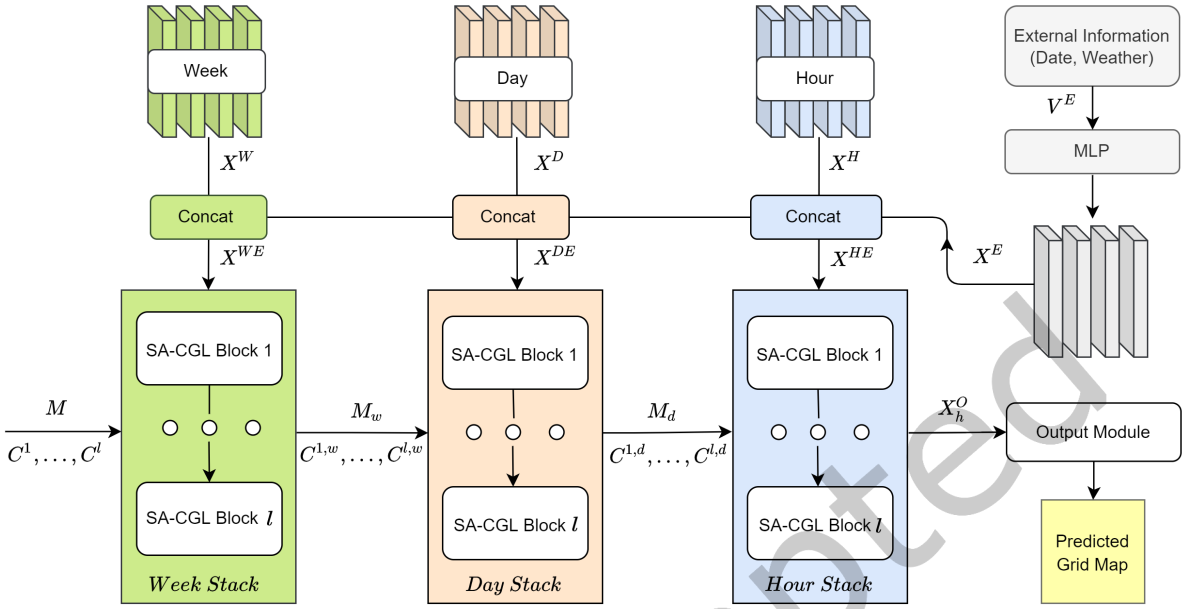

Fig. 2. The overall framework of DeepMeshCity. X^H, X^D, and X^W represent the historical citywide grid records of three fragments Hour, Day, and Week, respectively. V^E means the manually extracted external metadata, and X^E denotes the abstract representation for V^E after through two-layered MLP networks. Week Stack, Day Stack, and Hour Stack indicate shared stacked SA-CGL blocks. Week Stack takes as input the fused fragment X^{WE}, initial single-scale memory units $C^1, \ldots, C^l$, and multi-scale memory unit M. Then, the updated memory units $C^{1,w}, \ldots, C^{l,w}$ and M_w from the Week Stack are passed to the Day Stack as input. The same process is operated on the Hour Stack. The structures of the SA-CGL and stacked SA-CGL blocks are introduced in Figure 3 and Figure 4, respectively. The outcome X_h^O of the Hour Stack is conveyed to the output module to get the predicted grid map.

Instead of the late-fusion mechanism [48, 59], we adopt the early-fusion mechanism [12, 20] to concatenate the external metadata with each fragment tensor to yield the fused representations, that is,

$$X^{HE}, X^{DE}, X^{WE} = X^H \| X^E, X^D \| X^E, X^W \| X^E, \tag{2}$$

where $\|$ is the concatenation operation. Such a fusion allows our model to learn the dynamics by conditioning on different impacting factors and enables sharing the statistical strengths across similar conditions. The fused output X^{HE}, X^{DE}, and X^{WE} are then passed to the subsequent modules.

4.2 Self-Attention Citywide Grid Learner Block

The Self-Attention Citywide Grid Learner (SA-CGL) block is the core component of our proposed model. It consists of a self-attention unit and a Citywide Grid Learner (CGL) unit. The former aims to capture the global spatial dependencies and the latter is responsible for learning the spatial-temporal correlations by conditioning on the global spatial feature maps. For simplicity, we illustrate its design by only considering a single block in a given layer at a fixed time step.

4.2.1 Self-Attention Unit for Global Spatial Dependencies. The self-attention mechanism [29] enables each location's representation to be directly informed by the representations of all locations in the urban region, which

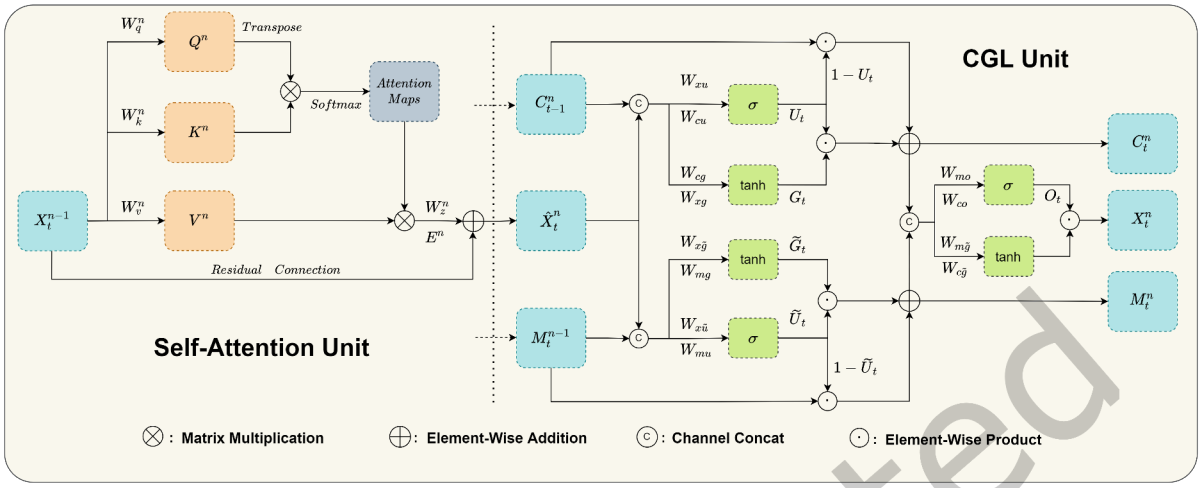

Fig. 3. The pipeline of the SA-CGL block. The left part is the self-attention unit while the other side shows the structure of the CGL unit. In the self-attention unit, X_t^{n-1} denotes the input feature maps for the n^{th} block at time step t. Q^n, K^n, and V^n represent query, key, and value based on 1×1 convolution on the feature maps, respectively. The output $\hat{X}_t^n$ is passed to the CGL unit. As for the CGL unit, besides $\hat{X}_t^n$, it takes a single-scale memory unit C_{t-1}^n at the previous moment, and a multi-scale memory unit M_t^{n-1} from the previous layer as input. Input modulation gates G_t, $\widetilde{G}_t$ and update gates U_t, $\widetilde{U}_t$ control the flow of spatial-temporal information for two memory units. Output gate O_t is used to select critical memory for the outcome. X_t^n, C_t^n, and M_t^n are the output for the block.

results in an effective global receptive field. In light of this property, we propose a self-attention unit to capture the global spatial dependencies for urban spatial-temporal data.

The left part of Figure 3 shows the pipeline of the proposed self-attention unit. The $X_t^{n-1} \in \mathbb{R}^{F_{in} \times H \times W}$ denotes the input feature maps for the n^{th} block at time step t, where F_{in} indicates the number of input channels. Firstly, X_t^{n-1} is mapped to query Q^n, key K^n, and value V^n by the convolution operation as follows:

$$\begin{aligned} Q^n &= W_q^n * X_t^{n-1} \in \mathbb{R}^{F_{out} \times N} \\ K^n &= W_k^n * X_t^{n-1} \in \mathbb{R}^{F_{out} \times N} \\ V^n &= W_v^n * X_t^{n-1} \in \mathbb{R}^{\hat{F}_{out} \times N}, \end{aligned} \tag{3}$$

where $\{W_q, W_k, W_v\}$ is a collection of weights for 1×1 convolution, F_{out} and $\hat{F}_{out}$ are output channels, and $N = H \times W$. The purpose of using 1×1 convolution is to ensure that the spatial structure of the urban feature maps can be preserved. The attention maps for pairwise points are further calculated by matrix multiplication and Softmax function as:

$$A^n = \text{Softmax}((Q^n)^T K^n) \in \mathbb{R}^{N \times N}. \tag{4}$$

The similarity scores between any two points can be obtained by indexing from the attention maps even if they are distant. If the similarity scores of two points are higher, their spatial dependencies are stronger, and vice versa. Then, the attention maps A^n are combined with values V^n via matrix multiplication and the Reshape operation, which can be summarized as:

$$E^n = \text{Reshape}(V^n A^n) \in \mathbb{R}^{\hat{F}_{out} \times H \times W}. \tag{5}$$

Each position aggregates its relationships by normalizing all positions in the corresponding row of attention maps, which allows the global spatial dependencies to be effectively captured. Later, we project the output to the same dimensions as the input and adopt a residual connection to stable the model training:

$$\hat{X}_t^n = W_z^n * E^n + X_t^{n-1} \in \mathbb{R}^{\hat{F}_{in} \times H \times W}. \tag{6}$$

Finally, the result $\hat{X}_t^n$ is passed to the CGL unit.

For the first block ($n = 1$), the self-attention unit learns the global spatial dependencies in the grid scale. Furthermore, it can obtain the global spatial dependencies in the area scale in the stacked blocks ($n > 1$), which will be elaborated on in the following subsection. The attention mechanism involves a $O(N^2)$ computational complexity, which might become a computation bottleneck when N is large. We will discuss how to reduce the complexity in Section 5.4.

4.2.2 Citywide Grid Learner Unit for Spatial-temporal Correlations. ConvLSTM [25] has been a popular model to represent spatial-temporal correlations. It has convolutional structures in both the input-to-state and state-to-state transitions, and it adopts a temporal memory state to learn the spatial-temporal correlations at the current scale. Although stacked ConvLSTMs can learn the correlations at different scales, they do not consider the multi-scale spatial-temporal correlations since temporal memory states are only updated along the time inside each ConvLSTM layer. Inspired by PredRNN [33, 34], our proposed citywide grid learner (CGL) unit employs a multi-scale memory unit to represent the multi-scale spatial-temporal correlations. Moreover, it can capture spatial-temporal correlations at different scales by the designed single-scale memory units.

As shown in the right part of Figure 3, the inputs of the CGL unit contain $\hat{X}_t^n$ from the self-attention unit, a single-scale memory unit C_{t-1}^n at the previous moment, and a multi-scale memory unit M_t^{n-1} from the previous layer. The computation of CGL can be summarized as follows:

$$\begin{aligned}
G_t &= \tanh(W_{xg} * \hat{X}_t^n + W_{cg} * C_{t-1}^n) \\
U_t &= \sigma(W_{xu} * \hat{X}_t^n + W_{cu} * C_{t-1}^n + B_c) \\
C_t^n &= (1 - U_t) \odot C_{t-1}^n + U_t \odot G_t \\
\widetilde{G}_t &= \tanh(W_{x\widetilde{g}} * \hat{X}_t^n + W_{mg} * M_t^{n-1}) \\
\widetilde{U}_t &= \sigma(W_{x\widetilde{u}} * \hat{X}_t^n + W_{mu} * M_t^{n-1} + B_m) \\
M_t^n &= (1 - \widetilde{U}_t) \odot M_t^{n-1} + \widetilde{U}_t \odot \widetilde{G}_t \\
O_t &= \sigma(W_{co} * C_t^n + W_{mo} * M_t^n) \\
X_t^n &= O_t \odot \tanh(W_{c\widetilde{g}} * C_t^n + W_{m\widetilde{g}} * M_t^n),
\end{aligned} \tag{7}$$

where $W_{\alpha\beta}$ ($\alpha \in \{x, c, m\}, \beta \in \{g, u, \widetilde{u}, \widetilde{g}, o\}$) denotes the learnable parameters in the CGL; B_c and B_m are constants; σ and tanh stand for the activation functions; $*$ represents the convolution operation; $\odot$ refers to the element-wise product. To lessen the computation and avoid overfitting, we leverage two activation functions to obtain input modulation gates G_t, $\widetilde{G}_t$ and update gates U_t, $\widetilde{U}_t$. Input modulation gates control the inflow of information when combining $\hat{X}_t^n$ with C_{t-1}^n and M_t^{n-1} at the current moment. U_t and $\widetilde{U}_t$ manage the dynamic changes of spatial-temporal information for memory units C_{t-1}^n and M_t^{n-1}. If U_t or $\widetilde{U}_t$ is close to 0, the memory unit tends to retain the past memory for urban characteristics. On the contrary, when the update gate is close to 1, it is more likely to learn the current spatial-temporal features. In this way, it enables an adaptive learning on the spatial-temporal correlations for C_t^n and M_t^n. In the end, we selectively control the outflow of two memory units via an output gate O_t.

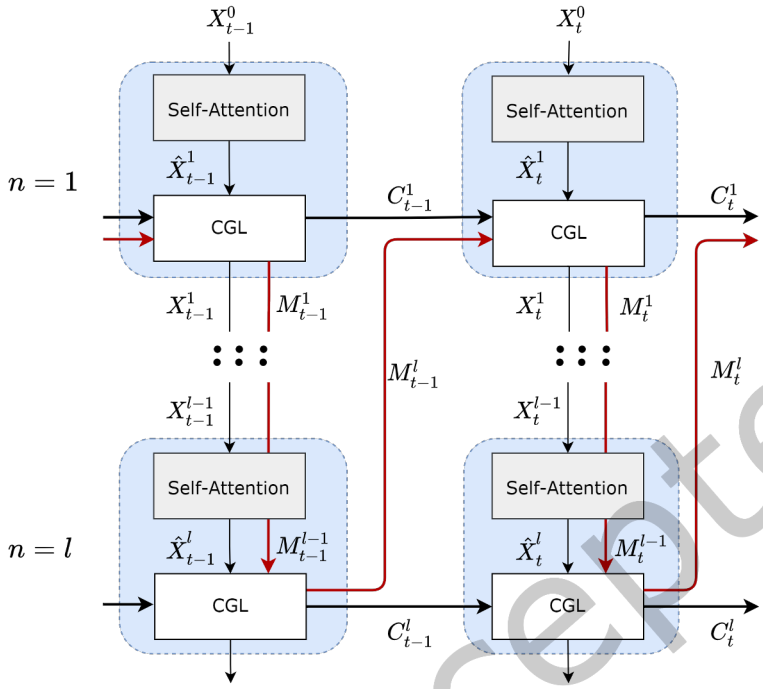

Fig. 4. The architecture of stacked SA-CGL blocks. The multi-scale memory unit M (red arrows) travels each CGL unit in all blocks in a zigzag path. Firstly, it flows downwards and across blocks at each time step to learn multi-scale spatial features. Then, M brings area-scale features in the top block to grid-scale characteristics in the bottom block at the next moment, to represent multi-scale spatial-temporal correlations.

Thus far, we only describe the mechanism of SA-CGL in a particular block at a fixed time step. To better represent the multi-scale spatial-temporal correlations, we now illustrate how to consecutively pass the M^n_t to the stacked blocks along the time in a zigzag path.

4.3 Stacked SA-CGL Blocks with the Multi-scale Memory Unit

As illustrated in Figure 4, we develop stacked SA-CGL blocks to capture the multi-scale spatial-temporal correlations. The multi-scale memory unit M (red line in Figure 4) traverses each CGL unit of all blocks along the time dimension. First of all, it flows downwards and across the blocks at each time step to learn multi-scale spatial features. Then, M brings area-scale features in the top block at the present step to grid-scale characteristics in the bottom block at the next moment, to represent multi-scale spatial-temporal correlations. In addition, we keep the grid area size constant to avoid the issue of semantic information loss, which the existing method [12] suffers when doing the downsampling operation.

Besides the multi-scale spatial-temporal correlations, the architecture could also model comprehensive global spatial dependencies and multiple single-scale spatial-temporal correlations at different scales. First, the self-attention unit can capture the global spatial dependencies at the grid scale from the urban grid maps X^0_t in the first block at the t timestamp. Then, the receptive fields of the output X^1_t are enlarged by 3×3 convolutions in the first CGL unit. Consequently, the subsequent self-attention units can thereby obtain the global spatial dependencies at

Table 1. The Description of Four Datasets.

Dataset	Grid Cell Size	Spatial Domain	Interval	Time Range	Data Shape
BousaiTYO Density	450m × 450m	80 × 80 grid cells	30 mins	04/01/2017 - 07/09/2017	(4800,80,80,1)
BousaiOSA Density		60 × 60 grid cells			(4800,60,60,1)
BousaiTYO Flow		80 × 80 grid cells			(4800,80,80,2)
TaxiBJ Traffic Flow	1000m × 1000m	32 × 32 grid cells	30 mins	07/01/2013 - 04/10/2016 Four Inconsecutive Parts	(20016,32,32,2)

an area-level scale. Meanwhile, we assign a single-scale memory unit to learn the spatial-temporal correlations in each SA-CGL block at the current scale. The stacked SA-CGL blocks include multiple single-scale memory units for different scales. Let $f_{\text{SA-CGL}}(\cdot)$ denote the transformation of CGL as shown in Equation 7, the computation of stacked SA-CGL blocks are presented as follows (for $2 \leq n \leq l$):

$$\begin{aligned} X_t^1, C_t^1, M_t^1 &= f_{\text{SA-CGL}_1}(X_t^0, C_{t-1}^1, M_{t-1}^l) \\ X_t^n, C_t^n, M_t^n &= f_{\text{SA-CGL}_n}(X_t^{n-1}, C_{t-1}^n, M_t^{n-1}), \end{aligned} \tag{8}$$

where l means the l^{th} block. The single-scale memory units $C_0^1, \ldots, C_0^n$ in each layer and the multi-scale memory M_0^1 are all initialized to zero tensor $\in \mathbb{R}^{F_h \times H \times W}$, where F_h is the feature dimension. Notably, for the first block ($n = 1$) at each time step, we set $M_t^{n-1} = M_{t-1}^l$.

As shown in Figure 2, the Week Stack, Day Stack, and Hour Stack represent the shared stacked SA-CGL blocks for different inputs in Equation 8. Each stack takes as input multiple single-scale memory units, the multi-scale memory unit, and the fused fragment in Equation 2. To speed up the model training, the memory units are carried from the previous fragment stack to the latter in our model. To illustrate that, for Week Stack, we feed the zero tensor to the memory units $C^1, \ldots, C^l$ and M to yield the output $C^{1,w}, \ldots, C^{l,w}$ and M_w by the SA-CGL blocks. The memory units keep the spatial-temporal features from weekly data X^{WE}. Then, we use $C^{1,w}, \ldots, C^{l,w}$ and M_w as the initial memory units for Day Stack. Thus, the memory units are able to obtain prior knowledge from previous ones. The same process is operated on the Hour Stack. Eventually, the aggregated feature X_h^O will be generated from the Hour Stack. We use f_{WS}, f_{DS}, and f_{HS} to represent the three fragment stacks, and the whole process can be summarized as:

$$\begin{aligned} X_w^O, C^{1,w}, \ldots, C^{l,w}, M_w &= f_{\text{WS}}(X^{\text{WE}}, C^1, \ldots, C^l, M) \\ X_d^O, C^{1,d}, \ldots, C^{l,d}, M_d &= f_{\text{DS}}(X^{\text{DE}}, C^{1,w}, \ldots, C^{l,w}, M_w) \\ X_h^O, C^{1,h}, \ldots, C^{l,h}, M_h &= f_{\text{HS}}(X^{\text{HE}}, C^{1,d}, \ldots, C^{l,d}, M_d). \end{aligned} \tag{9}$$

In Equation 9, X_h^O is the only result being passed to the next module. In the last step, we adopt an output module based on convolutions to obtain the outcome $\hat{Y}$ as:

$$\hat{Y} = W_{y2} * \text{LeakyReLU}(W_{y1} * X_h^O), \tag{10}$$

where W_{y1} and W_{y2} are learnable parameters of the convolution operation, and LeakyReLU is the activation function.

5 EXPERIMENT

In this section, we evaluate the ability of DeepMeshCity for urban grid prediction on two typical tasks: crowd density prediction and flow prediction. The experiments are conducted on four real-world urban datasets—BousaiTYO and BousaiOSA crowd density datasets, BousaiTYO crowd flow dataset, and TaxiBJ traffic flow dataset. The details of datasets are summarized in Table 1.

5.1 Dataset

TaxiBJ is one of the most widely used traffic flow datasets in the literature [48]. The dataset records the GPS coordinates of the taxicab in Beijing during four time periods, 07/01/2013-10/30/2013, 03/01/2014/-06/30/2014, 03/01/2015-06/30/2015, and 11/01/2015-04/10/2016. The sampling time interval is 30 minutes and the entire period covers 18 months. We partition the city into 32×32 grid cells with each grid cell size 1000m $\times$ 1000m, which yields a tensor of size $20016 \times 32 \times 32 \times 2$.

BousaiTYO and BousaiOSA. Bousai datasets are released by Yahoo Japan Corporation [11]. The dataset records the location information of millions of users in Japan with a sampling interval of 30 minutes. The records of two big cities (Tokyo and Osaka) from 1 April 2017 to 9 July 2017 (100 days) are selected in our experiments. We refer to the corresponding datasets as BousaiTYO and BousaiOSA, respectively. The two cities are partitioned into 80×80 and 60×60 grid cells, respectively, with a grid cell size 450m $\times$ 450m. Consequently, the BousaiTYO dataset contains a crowd density tensor of size $4800 \times 80 \times 80 \times 1$ and a crowd flow tensor of size $4800 \times 80 \times 80 \times 2$, whereas the BousaiOSA dataset only contains a crowd density tensor of size $4800 \times 60 \times 60 \times 1$.

5.2 Baseline Methods

We evaluate the performance of our proposed method by comparing it with the grid-based baseline methods:

- HistoricalAverage. We use the average of the historical values from the corresponding timestamp.
- CopyYesterday. We directly copy the value of the corresponding timestamp in the last day as the predicted value.
- CopyLastFrame. We adopt the most recent observation as the predicted value.
- CNN is the vanilla convolutional neural network. The input tensor is concatenated along the time dimension to yield a tensor of shape $(H, W, T * F)$. The model employs four convolutional layers with 32 filters of 3×3 kernel size.
- ConvLSTM is a spatial-temporal forecasting model proposed by [25]. The network utilizes four ConvLSTM layers with 32 filters of 3×3 kernel size.
- ST-ResNet [48]. Deep Spatio-Temporal Residual Network (ST-ResNet) is the most representative grid-based method which designs a residual convolutional framework for crowd flow prediction. In the study, two residual units are adopted.
- DMVST-Net [39]. Deep Multi-View Spatial-Temporal Network (DMVST-Net) is a typical deep learning framework for taxi demand prediction based on CNN and LSTM. It adopts 9 neighbor grid cells as the input of local CNN. The graph embedding is 32 and the temporal output is 512 for LSTM.
- PCRN [59]. Convolutional Recurrent Network with Periodic Representation (PCRN) is a ConvGRU-based model for taxi density prediction. It builds a pyramidal architecture with three stacked ConvGRU layers.
- STDN [38]. Spatial-Temporal Dynamic Network (STDN) is an improved version of DMVST-Net. It adopts a flow gating mechanism to fuse flow information of 9 local grid cells. The hidden output of the periodically shifted attention mechanism is 128.
- DeepSTN+ [20]. Context-aware Spatial-Temporal Neural Network (DeepSTN+) is an improved version of ST-ResNet for crowd flow prediction. We use 2 ResPlus units and the ConvPlus channel, and the separated channels are set as 32 and 8, respectively.

- DeepCrowd [12] is a crowd density prediction model based on ConvLSTM, which is equipped with the pyramid architecture and attention mechanism. It consists of three bottom-up ConvLSTM layers with {32, 64, 128} filters of 3×3 kernel size and three top-down ConvLSTM layers with {128, 128, 128} filters of 1×1 kernel size.

5.3 Experimental Setup

All experiments are conducted on a GeForce GTX 1080 Ti GPU. We use the Adamw optimizer with a learning rate of 0.001 and weight decay of 5×10^{-4}, and the cosine annealing strategy [22] is adopted to decay the learning rate. We adopt the Min-Max normalization method to scale the data into the range [0, 1] before feeding it into the model, and then rescale the predicted value back to the original value. The training process is carried out for 155 epochs, with a batch size of 4 samples. Early stopping is used on the validation dataset to select the best model. The sub-scaling factor for each self-attention unit is 2. In this paper, we apply a mean square error (MSE) between the estimator and the ground truth as the loss function. Each fragment stack contains two SA-CGL blocks with {64,64} filters for the self-attention units and CGL units. The stacked SA-CGL blocks are shared among the fragment stacks.

TaxiBJ. The observation steps are set as follows: $T_h = 6$, $T_d = 1$, and $T_w = 1$. It means if we aim to predict the traffic flow of Beijing from 6:30 p.m. to 7:00 p.m. on 03/30/2016, then the input is X^H: 03/30/2016 4:00 p.m. ~ 6:30 p.m., X^D: 03/29/2016 6:30 p.m. ~ 7:00 p.m., and X^W: 03/23/2016 6:30 p.m. ~ 7:00 p.m.. The data ratio for training, validation, and testing is set as 7:1:2. External data V^E include holidays (1 feature), meteorology data (19 features), and meta-date (8 features).

BousaiTYO and BousaiOSA. We select data from the first 80% as training data (20% of which are taken as validation data), and the remaining 20% are set as testing data. The observation step T is 6. It indicates that if we want to forecast the crowd density of Tokyo from 6:30 p.m. to 7:00 p.m. on 04/30/2017, then the input is X^H: 03/30/2017 4:00 p.m. ~ 6:30 p.m., X^D: 03/29/2016 4:00 p.m. ~ 6:30 p.m., and X^W: 03/23/2016 4:00 p.m. ~ 6:30 p.m. Only meta-date will be used as external information, which contains the time of the day (48 features), days of the week (7 features), weekday or not (1 feature), and holiday or not (1 feature).

5.4 Preprocessing

We design two preprocessing strategies to tackle the problems of input selection and the complexity of self-attention.

Input selection. It aims to select the lengths of X^W, X^D, and X^H mentioned in Section 4.1. The selection of the three fragments is different for Bousai datasets and TaxiBJ. We follow the setups from DeepCrowd [12] for Bousai-related datasets. Intuitively, the three parts of the historical observations can be represented as:

$$\begin{aligned} X^H &= [X_{t-T}, X_{t-(T-1)}, \ldots, X_{t-1}] \\ X^D &= [X_{(t-f_d)-T}, X_{(t-f_d)-(T-1)}, \ldots, X_{(t-f_d)-1}] \\ X^W &= [X_{(t-f_w)-T}, X_{(t-f_w)-(T-1)}, \ldots, X_{(t-f_w)-1}], \end{aligned} \tag{11}$$

where T is the observation step in each fragment, f_d and f_w are sampling frequencies during a day and a week, which are 48 and 7×48 respectively since the time interval of Bousai datasets is 30 minutes. As for TaxiBJ, we adopt the setting from ST-ResNet [48]. Then, the selection of three fragments can be expressed as:

$$\begin{aligned} X^H &= [X_{t-T_h}, X_{t-(T_h-1)}, \ldots, X_{t-1}] \\ X^D &= [X_{t-T_d \times f_d}, X_{t-(T_d-1)\times f_d}, \ldots, X_{t-f_d}] \\ X^W &= [X_{t-T_w \times f_d}, X_{t-(T_w-1)\times f_d}, \ldots, X_{t-f_w}], \end{aligned} \tag{12}$$

where T_h, T_d, and T_w stand for the lengths of the most recent observations, daily periodicity, and weekly trend separately, f_d and f_w are 48 and 7×48 too because their time interval is also 30 minutes. It is notable that our model is capable of performing predictions at various time granularity.

Sub-scaling for self-attention unit. The self-attention mechanism suffers from high computational complexity given the large mesh-grid number since it calculates the similarity among all points [21]. Inspired by the sub-scaling method [25, 33], we propose to transform the spatial-temporal input from $F \times H \times W$ to $(F \cdot P \cdot P) \times (H/P) \times (W/P)$ for each self-attention unit, where P is the sub-scaling factor. It will be reshaped to the original shape before passing to the CGL unit. In such a manner, we can reduce both the computational and memory complexity without compromising the accuracy.

5.5 Evaluation Metric

Mean Square Error(MSE), Root Mean Square Error (RMSE), and Mean Absolute Error (MAE) are evaluation metrics:

$$\begin{aligned} \text{MSE} &= \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \\ \text{RMSE} &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2} \\ \text{MAE} &= \frac{1}{n}\sum_{i=1}^{n} \mid Y_i - \hat{Y}_i \mid, \end{aligned} \tag{13}$$

where Y_i and $\hat{Y}_i$ represent the ground truth and predicted grid map, respectively, and n is the number of all predicted values. We select MSE and MAE for Bousai dataset, and the RMSE and MAE for TaxiBJ, for a fair comparison in evaluation protocols with the previous works.

5.6 Results and Analysis on Crowd Density Prediction

For the crowd density prediction task, the performance of our proposed model is compared with the relevant baseline methods on BousaiTYO and BousaiOSA datasets. Table 2 summarizes the results, with benchmarking the MSE and MAE results of ST-ResNet, which are indicated by the relative increments, Δ MSE and Δ MAE. We observe that our DeepMeshCity achieves the best performance on both MSE and MAE metrics when a relatively small number of parameters are used.

Among all baseline methods, the simple statistical methods (HistoricalAverage and CopyYesterday) perform worst. The naive CNN reduces the MSE and MAE errors by approximately 44.72% and 20.92% in comparison to the HistoricalAverage. Because the former has the ability to learn the non-linear features from the data, while traditional methods are limited to capturing spatial-temporal characteristics from complex urban data. Another deep learning model, ConvLSTM is better than CNN since it can capture both the spatial and temporal information rather than merely considering the spatial features, by extending the fully connected LSTM to the convolutional structures. However, the aforementioned deep learning approaches are not good to model complex spatial-temporal correlations.

As for other deep learning approaches, even though they manage to capture more comprehensive spatial-temporal correlations, their performance is still under-expected. ST-ResNet fails to capture the global spatial dependencies and spatial-temporal correlations efficiently, although it achieves a decent result with the superiority of parameter efficiency. The inefficiency results from its stacked CNN-based residual units and the way to concatenate the channel dimension with the time dimension. The updated version Deep-STN+, aiming at efficiency improvement on global spatial dependency capture, performs poorly too. Because the loss of POIs

Table 2. Performance comparison of various models on BousaiTYO and BousaiOSA crowd density datasets for crowd density prediction.

	Bousai Tokyo Crowd Density					Bousai Osaka Crowd Density			
Models	**#Params**	**MSE ↓**	**ΔMSE**	**MAE ↓**	**Δ MAE**	**MSE ↓**	**Δ MSE**	**MAE ↓**	**Δ MAE**
CopyYesterday	-	1304.393	2394.63%	11.804	166.09%	274.857	1195.15%	7.264	150.48%
HistoricalAverage	-	225.501	323.62%	7.715	61.74%	79.557	274.88%	4.864	67.73%
CNN	-	124.657	138.40%	6.101	37.53%	35.325	66.45%	3.203	10.44%
ConvLSTM [25]	0.18M	81.778	56.40%	4.335	-2.27%	26.939	26.94%	2.940	1.38%
ST-ResNet [48]	0.20M	52.288	0.00%	4.336	0.00%	21.222	0.00%	2.900	0.00%
DMVST-Net [39]	1.58M	42.726	-18.28%	3.918	-11.67%	17.852	-15.88%	2.613	-9.89%
PCRN [59]	1.28M	55.676	6.48%	4.653	4.89%	21.064	-0.74%	2.898	-0.07%
STDN [38]	6.18M	39.492	-24.47%	3.713	-16.30%	22.791	7.39%	2.884	-0.55%
DeepSTN+ [20]	327.82M	89.775	71.69%	4.907	10.61%	32.962	55.32%	3.230	11.38%
DeepCrowd [12]	7.88M	33.138	-36.62%	3.394	-23.49%	16.743	-21.10%	2.458	-15.24%
DeepMeshCity	0.98M	29.670	-43.25%	3.228	-27.23%	15.445	-27.22%	2.383	-17.83%

information on Bousai datasets leads to its weak effect on capturing long-range spatial dependencies in the whole city. In particular, the fully-connected operation in its ConvPlus block results in huge numbers of parameters in BousaiTYO, affecting the model efficiency.

To better represent the spatial-temporal correlations, DMVST-Net and STDN are developed and achieve better results than ST-ResNet, by adopting the local CNN and LSTM models. However, these trials neglect the multi-scale spatial-temporal correlations, preventing them from higher achievements. In contrast, PCRN and DeepCrowd both employ a pyramidal ConvLSTM-based model for these correlations. PCRN achieves better results on BousaiOSA but is worse on BousaiTYO than ST-ResNet. Because the BousaiTYO dataset has a much larger mesh-grid number, this makes PCRN very inefficient in dynamically saving the updated periodic representation for each step. In contrast, DeepCrowd attains the second-best results on both datasets owing to its upsampling operation which fuses area-scale representation with grid-scale features. However, as we mentioned in Section 1 the area-scale semantic information might not propagate to grid-scale features effectively by upsampling due to its coarse interpolation.

Our DeepMeshCity demonstrates its superiority in comparison to the baselines in terms of both MSE and MAE on both datasets with a relatively small model scale. It introduces the SA-CGL block to capture the global spatial dependencies and uses stacked SA-CGL blocks with a multi-scale memory unit to represent the multi-scale spatial-temporal correlations. Compared with ST-ResNet, DeepMeshCity contributes to the considerable decrease of MSE and MAE by 43.25% and 27.23% on BousaiTYO, and 27.22% and 17.83% on BousaiOSA. Besides, it surpasses the state-of-the-art method DeepCrowd by 10.46% in terms of MSE and 4.89% in terms of MAE (7.75%, 3.05%) on BousaiTYO (BousaiOSA). These results show that our model has more advantages in expressing the global spatial dependencies and the multi-scale spatial-temporal correlations of crowd density.

5.6.1 Efficiency and Scalability. In addition to the comparison of model parameter, we also provide a comparison of different models in terms of computational time and memory usage on the BousaiTYO and BousaiOSA datasets, as these metrics are crucial for practical deployment. First, we plot the training time for one epoch in minutes and memory usage for the typical models in Figure 5(a) and Figure 5(b), respectively. From the figures, we can

observe that: (1) ST-ResNet has a significant advantage in computational efficiency and memory usage compared to other models on both datasets. However, the manner of concatenating feature and temporal dimensions (to apply CNN) hinders its real-time dependency capturing ability, thus limiting scalability; (2) STDN employs local CNN to treat each grid cell as a computational unit, resulting in the low efficiency on training time (nearly 5 days on BousaiTYO). Nevertheless, it only involves neighboring $n * n$ grids in the computation, leading to a relatively small memory usage that does not vary with urban size. (3) DeepCrowd stacks three bottom-up ConvLSTM layers and three top-down ConvLSTM layers by the proposed Prmarid ConvLSTM, consequently leading to larger memory usage and longer computation time. (4) Our model performs well in terms of computational efficiency and scalability, mainly due to the efficient design of the CGL module and the sub-scaling method which significantly reduces the memory usage of the self-attention module. In summary, we can recommend DeepMeshCity as a viable solution to the real-world prediction tasks of urban crowds and traffic with a good balance of efficiency and scalability.

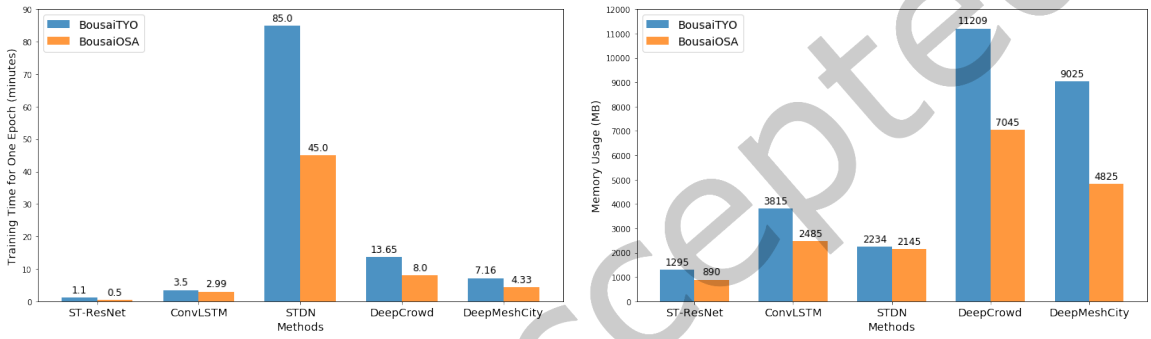

Fig. 5. The training time for each epoch in minutes and memory usage for different models on BousaiTYO and BousaiOSA.

5.6.2 Ablation Experiments. To verify the effectiveness of different components proposed in DeepMeshCity, we conduct ablation experiments on the BousaiTYO crowd density dataset in this section. To this end, six variants of DeepMeshCity are designed in these experiments, namely, 1) DeepMeshCity-noSA: it removes the self-attention unit for each SA-CGL block; 2) DeepMeshCity-noM: it eliminates the multi-scale memory unit; 3) DeepMeshCity-noM-noSA: it excludes both the self-attention unit and the multi-scale memory unit; 4) DeepMeshCity-CGLtoSTLSTM: it replaces the CGL unit with the ST-LSTM unit [33]; 5) DeepMeshCity-SeparateStack: it substitutes three shared fragment stacks with three separated stacks. All other components remain fixed except the aforementioned modules.

As shown in Table 3, DeepMeshCity-noSA and DeepMeshCity-noM perform worse than DeepMeshCity. It indicates that the proposed self-attention unit and the multi-scale memory unit are critical, and the global spatial dependencies and multi-scale spatial-temporal correlations are conducive to the overall performance. The weaker prediction effect of DeepMeshCity-noM-noSA further supports the conclusion. The DeepMeshCity-CGLtoSTLSTM gives rise to the worst results. We hypothesize that this is due to the ST-LSTM unit being much more complex than our CGL unit, it is prone to overfitting and suffering from the gradient vanishing problem. The last model in the table DeepMeshCity-SeparateStack demonstrates a competitive performance, but it uses more parameters than ours. It implies that our proposed model possesses the ability in learning spatial-temporal features with the shared SA-CGL blocks. The experiments demonstrate both the effectiveness and reliability of every module of our proposed model.

Table 3. Performance comparison of various components of DeepMeshCity on BousaiTYO crowd density.

	Bousai Tokyo Crowd Density		
Variant Models	**#Params**	**MSE ↓**	**MAE↓**
DeepMeshCity-noSA	0.91M	30.077	3.250
DeepMeshCity-noM	0.83M	31.359	3.377
DeepMeshCity-noM-noSA	0.76M	31.956	3.378
DeepMeshCity-CGLtoSTLSTM	1.37M	32.584	3.373
DeepMeshCity-SeparateStack	1.95M	29.861	3.228
DeepMeshCity	0.98M	29.670	3.228

5.6.3 Effect of Different Network Configurations. Figure 6 presents the influence of the different network configurations on BousaiTYO crowd density. Next, we study the impact of two hyper-parameters: output channel size and block depth.

Channel of the SA-CGL. The model performance increases when the output channel size of SA-CGL block grows from 16 to 64, and it drops when the channel size increases to 128. This shows that a larger model owns more capacity in learning the spatial-temporal patterns but can be overfitting if the model is too large.

Depth of the SA-CGL. In Figure 6, network depth 2 is the turning point of the model performance. When the network depth is 1, the model performance is not as competitive as 2, since the model only considers the spatial-temporal correlations of the grid scale regardless of the area scale. No performance gain is observed when the depth is larger than 2. We hypothesize that it is caused by the training difficulty of deep architecture. Thus, our selection of network configurations is reasonable with an output channel size of 64 and block depth of 2.

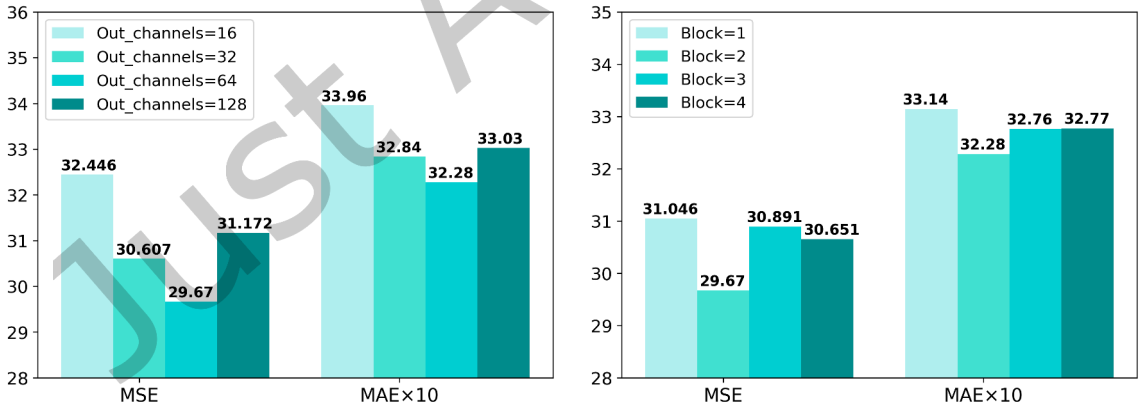

Fig. 6. The performance of different network configurations on BousaiTYO crowd density, including the channel size of the SA-CGL block (left part) and the depth of SA-CGL blocks (right part). The MAE is expanded by ten times for a better presentation.

Table 4. Performance comparison of various models on BousaiTYO crowd flow.

	Bousai Tokyo Crowd Flow			
Models	**MSE ↓**	**Δ MSE**	**MAE ↓**	**Δ MAE**
CopyYesterday	148.604	592.18%	4.156	68.46%
HistoricalAverage	47.433	120.94%	3.090	25.25%
CNN	49.070	128.56%	3.812	54.52%
ConvLSTM [25]	24.237	12.89%	2.540	2.96%
ST-ResNet [48]	21.469	0.00%	2.467	0.00%
DMVST-Net [39]	34.795	62.07%	2.985	20.99%
PCRN [59]	22.710	5.78%	2.491	0.97%
STDN [38]	19.654	-8.45%	2.468	0.04%
DeepSTN+ [20]	19.062	-11.21%	2.387	-3.24%
DeepCrowd [12]	18.697	-12.91%	2.203	-10.70%
DeepMeshCity	17.999	-16.16%	2.185	-11.43%

Table 5. Performance comparison of various models on TaxiBJ traffic flow.

	TaxiBJ Traffic Flow			
Models	**RMSE ↓**	**ΔRMSE**	**MAE ↓**	**Δ MAE**
HistoricalAverage	45.004	140.63%	24.475	133.25%
CopyLastFrame	23.609	26.23%	13.372	27.43%
CNN	23.550	25.92%	13.797	31.48%
ConvLSTM [25]	19.247	2.91%	10.816	3.08%
ST-ResNet [48]	18.702	0.00%	10.493	0.00%
DMVST-Net [39]	20.389	9.02%	11.832	12.76%
PCRN [59]	18.629	-0.39%	10.432	-0.58%
DeepSTN+ [20]	18.141	-3.00%	10.126	-3.49%
STDN [38]	17.826	-4.68%	9.901	-5.64%
DeepMeshCity	16.895	-9.66%	9.627	-8.25%

5.7 Results and Analysis on Flow Prediction

For flow prediction, we conduct comparative experiments of our model on BousaiTYO crowd flow and TaxiBJ traffic flow datasets. DeepMeshCity maintains its dominance in all metrics on both datasets with the proposed SA-CGL blocks.

As shown in Table 4, most models on BousaiTYO crowd flow intuitively have similar comparison results as those of BousaiTYO crowd density. However, DeepSTN+ and DMVST-Net show a reverse tendency on BousaiTYO crowd flow when compared to ST-ResNet. Compared with static crowd density, the crowd in-out flow is dynamic and sparse. To illustrate that, considering the midnight, the crowd density maps keep a relatively high density while the crowd in-out flow heatmap might be very sparse since few people are walking around. Thus, DeepSTN+

shows better performance on the crowd flow task whereas DMVST-Net suffers when the input data becomes sparser. Because the former model is able to capture the long-range spatial dependencies more easily with its ConvPlus block, while the latter only takes local grid cells as input rather than global grid cells. When the data is sparse, there will be no flow on most local grid cells, which has a negative effect on the model training.

In the TaxiBJ traffic flow dataset depicted in Table 5, there is little difference in the performance of the deep learning methods (ConvSTM ~ STDN), but the overall performance is much worse than the BousaiTYO crowd flow. We conjecture the underlying reason is that the accurate spatial correlations are hard to capture on TaxiBJ traffic flow, with its smaller spatial domain with 32×32 grid cells than BousaiTYO crowd flow. However, even in this case, our approach outperforms other models and outperforms STDN (ST-ResNet) by 5.22% in RMSE, and 2.76% in MAE (9.66%, 8.25%). In summary, the experimental results show that our proposed DeepMeshCity is applicable to both large and small mesh-grid number datasets for flow prediction.

5.8 Case Study

In this section, we analyze the spatial attention weight map learned by the proposed self-attention mechanism on BousaiTYO crowd density to demonstrate the effectiveness of capturing global spatial patterns.

We select the Tokyo Station grid cell (35, 53), which contains the largest station in Tokyo, as our case study. Figure 8 is the normalized attention map of the Tokyo Station grid in the top layer of the SA-CGL from 6:00 p.m. to 6:30 p.m. We can observe that there is an expected high correlation between the Tokyo Station and the surrounding vicinity of area **S**. It is due to the likelihood of individuals commuting to work, taking leisurely strolls, having dinner, or preparing to return home from nearby places. The region **A** exhibits a notably high level of correlation. The specific area corresponds to the Shin-Yokohama Station grid cell (77, 25) and the encompassing grid cells form a residential area. Despite the considerable distance from the Tokyo Station, it remains closely connected due to direct access. Similarly, the region **B** contains the Kawasaki Station grid cell (72, 39) and demonstrates the same pattern. The region **C** corresponds to the Tokyo International Airport grid cell (67, 57), a common destination for individuals commuting via the subway. Hence, this grid area indicates a distinctive and heightened correlation with the Tokyo Station. The above analysis of the attention map for the Tokyo Station grid shows the efficacy of the proposed self-attention module in capturing comprehensive spatial patterns in urban areas.

5.9 Visualization Analysis

We briefly demonstrate the prediction results and corresponding error histogram of DeepMeshCity for the above four datasets. We only present the partial recent input sequence instead of all inputs due to the limited space. Figures 7 and 9 back up the superiority of our model. In Figure 7, our model produces a very accurate prediction over the whole grid cells of Tokyo and Osaka during working time and off time, especially for the changes in the hot area. Additionally, the prediction error of most grid cells for these two cities in Figure 9 is less than 10 on Bousai datasets but is much larger in TaxiBJ traffic flow. The underlying reason could be that the flow variation within a time interval is larger and it is more difficult to predict precisely in TaxiBJ, whose spatial region is downtown. The differences between ConvLSTM and our model are visualized in Figure 10. It can be seen that our model obtains better achievement. Therefore, DeepMeshCity is a general framework not only suitable for crowd density prediction but also excels at flow prediction.

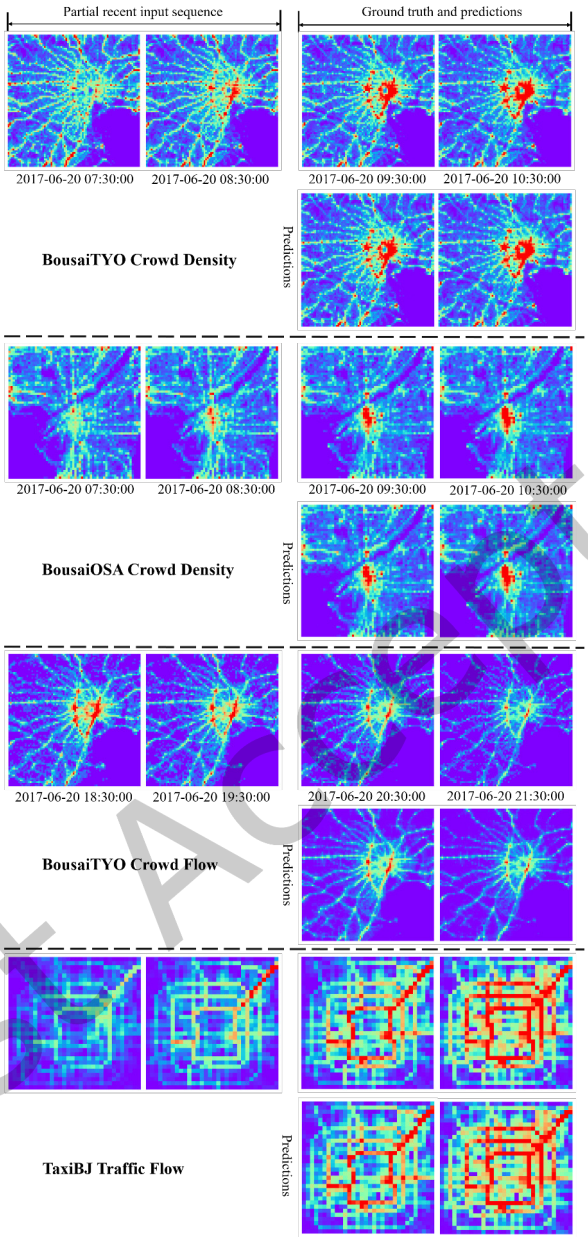

Fig. 7. Prediction visualization of DeepMeshCity on four datasets. We select the inflow for crowd/traffic flow prediction.

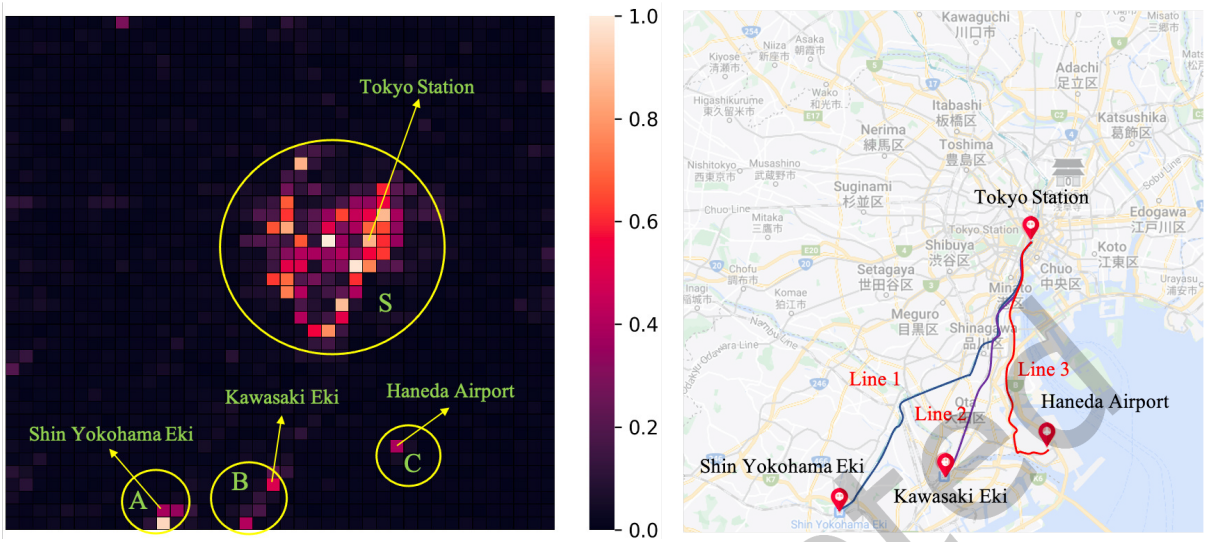

Fig. 8. Case study of the spatial attention map. The Tokyo Station grid cell (35, 53) is selected.

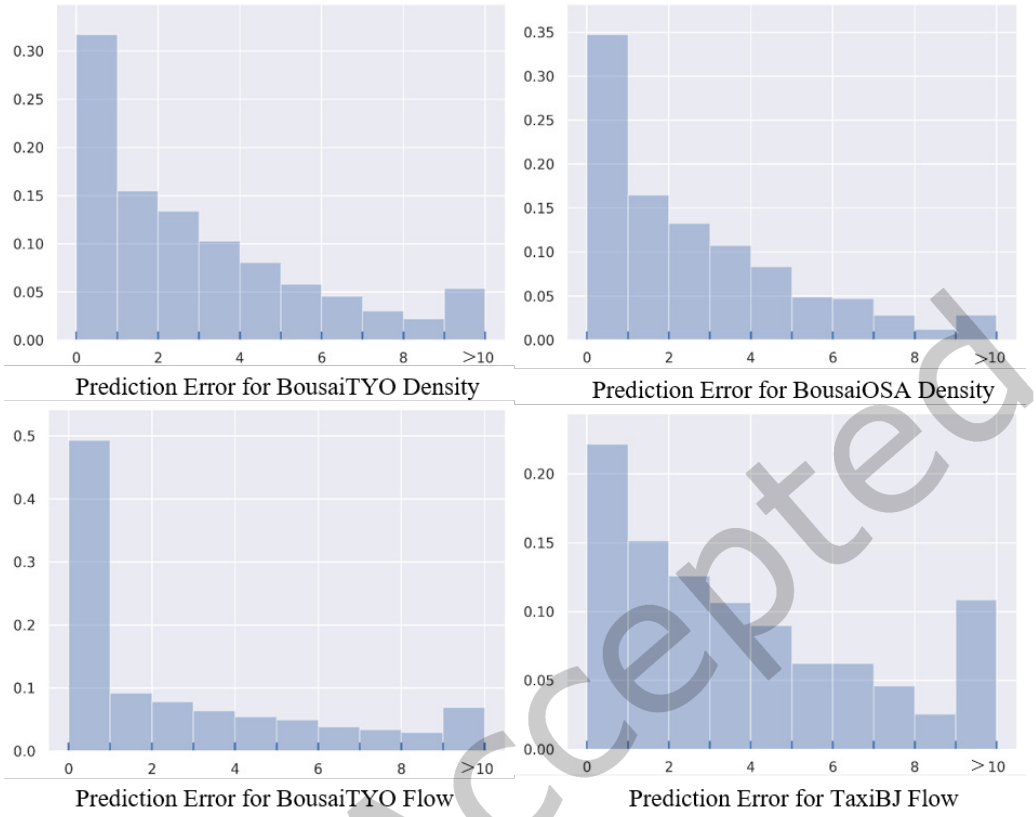

Fig. 9. Error histogram of DeepMeshCity for ground truth and prediction on four datasets. We present the rightmost predictions in Figure 7.

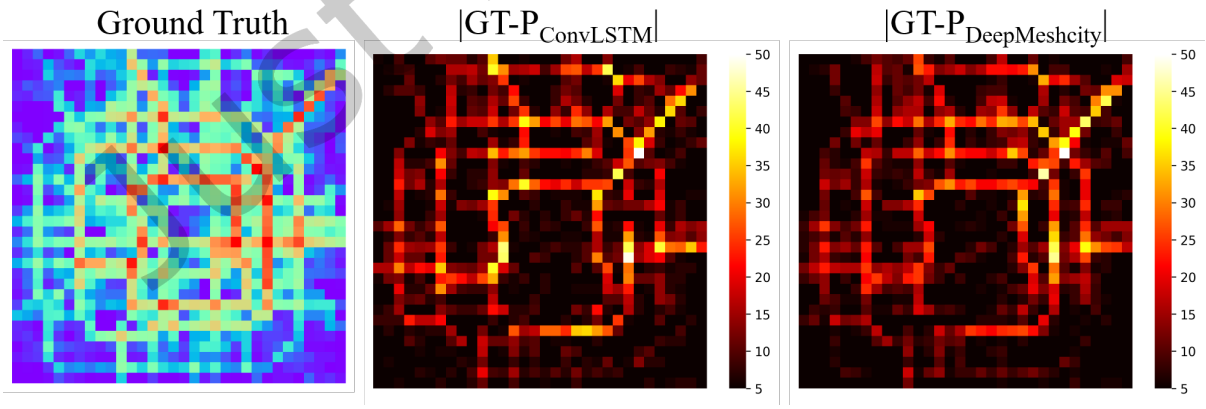

Fig. 10. Visualization of the prediction difference for ConvLSTM and DeepMeshCity on TaxiBJ. P_{ConvLSTM} and $P_{\text{DeepMeshCity}}$ indicate the predictions of ConvLSTM and DeepMeshCity respectively.

6 CONCLUSION

In this paper, we propose a deep learning framework DeepMeshCity for urban grid prediction. DeepMeshCity can capture the global spatial dependencies and the multi-scale spatial-temporal correlations effectively with the proposed SA-CGL block and the multi-scale memory unit. The experimental results on four real-world datasets validate that our method is applicable to both crowd density prediction and crowd/traffic flow prediction regardless of the grid scale and cell size. In addition to traffic management, our model could also contribute to citywide development by guiding the placement of new residential, commercial, and public facilities through flow distribution analysis to ensure high accessibility for residents. Moreover, in community vitality assessment [31], it leverages social interaction data from crowd flow shifts to optimize the layout of POIs such as commercial areas and transportation stations around communities to enhance their vibrancy. However, we notice that the multi-scale memory unit may suffer from gradient propagation problems with the deep architecture, which requires further exploration in the future.

ACKNOWLEDGMENTS

The authors would like to thank Yahoo Japan Corporation for providing the Bousai crowd data. This work is supported by the National Natural Science Foundation of China under Grant No.:~62206074 and the Shenzhen College Stability Support Plan under Grant No.:~GXWD20220811173233001.

REFERENCES

[1] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. 2020. Adaptive graph convolutional recurrent network for traffic forecasting. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada) *(NIPS'20)*. Curran Associates Inc., Red Hook, NY, USA, Article 1494, 12 pages.

[2] Ling Cai, Krzysztof Janowicz, Gengchen Mai, Bo Yan, and Rui Zhu. 2020. Traffic transformer: Capturing the continuity and periodicity of time series for traffic forecasting. *Transactions in GIS* 24, 3 (2020), 736–755. https://doi.org/10.1111/tgis.12644 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/tgis.12644

[3] Weiqi Chen, Ling Chen, Yu Xie, Wei Cao, Yusong Gao, and Xiaojie Feng. 2020. Multi-Range Attentive Bicomponent Graph Convolutional Network for Traffic Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 04 (Apr. 2020), 3529–3536. https://doi.org/10.1609/aaai.v34i04.5758

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, and Sylvain Gelly. 2020. An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv preprint arXiv:2010.11929* (2020).

[5] Bowen Du, Hao Peng, Senzhang Wang, Md Zakirul Alam Bhuiyan, Lihong Wang, Qiran Gong, Lin Liu, and Jing Li. 2019. Deep Irregular Convolutional Residual LSTM for Urban Traffic Passenger Flows Prediction. *IEEE Transactions on Intelligent Transportation Systems* 21, 3 (2019), 972–985. https://doi.org/10.1109/TITS.2019.2900481

[6] Xu Geng, Yaguang Li, Leye Wang, Lingyu Zhang, Qiang Yang, Jieping Ye, and Yan Liu. 2019. Spatiotemporal Multi-Graph Convolution Network for Ride-Hailing Demand Forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 3656–3663. https://doi.org/10.1609/aaai.v33i01.33013656

[7] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. 2019. Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 922–929.

[8] Chao Huang, Chuxu Zhang, Jiashu Zhao, Xian Wu, Dawei Yin, and Nitesh Chawla. 2019. MiST: A Multiview and Multimodal Spatial-Temporal Learning Framework for Citywide Abnormal Event Forecasting. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 717–728. https://doi.org/10.1145/3308558.3313730

[9] Jiawei Jiang, Chengkai Han, Wayne Xin Zhao, and Jingyuan Wang. 2023. PDFormer: propagation delay-aware dynamic long-range transformer for traffic flow prediction. , Article 487 (2023), 9 pages. https://doi.org/10.1609/aaai.v37i4.25556

[10] Renhe Jiang, Zekun Cai, Zhaonan Wang, Chuang Yang, Zipei Fan, Quanjun Chen, Xuan Song, and Ryosuke Shibasaki. 2022. Predicting Citywide Crowd Dynamics at Big Events: A Deep Learning System. 13, 2, Article 21 (mar 2022), 24 pages. https://doi.org/10.1145/3472300

[11] Renhe Jiang, Zekun Cai, Zhaonan Wang, Chuang Yang, Zipei Fan, Quanjun Chen, Kota Tsubouchi, Xuan Song, and Ryosuke Shibasaki. 2022. Yahoo! Bousai Crowd Data: A Large-Scale Crowd Density and Flow Dataset in Tokyo and Osaka. In *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 6676–6677. https://doi.org/10.1109/BigData55660.2022.10020886

[12] Renhe Jiang, Zekun Cai, Zhaonan Wang, Chuang Yang, Zipei Fan, Quanjun Chen, Kota Tsubouchi, Xuan Song, and Ryosuke Shibasaki. 2023. DeepCrowd: A Deep Model for Large-Scale Citywide Crowd Density and Flow Prediction. *IEEE Transactions on Knowledge & Data Engineering* 35, 1 (2023), 276–290. https://doi.org/10.1109/TKDE.2021.3077056

[13] Renhe Jiang, Xuan Song, Dou Huang, Xiaoya Song, Tianqi Xia, Zekun Cai, Zhaonan Wang, Kyoung-Sook Kim, and Ryosuke Shibasaki. 2019. DeepUrbanEvent: A System for Predicting Citywide Crowd Dynamics at Big Events. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 2114–2122. https://doi.org/10.1145/3292500.3330654

[14] Renhe Jiang, Du Yin, Zhaonan Wang, Yizhuo Wang, Jiewen Deng, Hangchen Liu, Zekun Cai, Jinliang Deng, Xuan Song, and Ryosuke Shibasaki. 2021. DL-Traff: Survey and Benchmark of Deep Learning Models for Urban Traffic Prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (Virtual Event, Queensland, Australia) *(CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 4515–4525. https://doi.org/10.1145/3459637.3482000

[15] Thomas N Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv preprint arXiv:1609.02907* (2016).

[16] Mengzhang Li and Zhanxing Zhu. 2021. Spatial-Temporal Fusion Graph Neural Networks for Traffic Flow Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 5 (May 2021), 4189–4196. https://doi.org/10.1609/aaai.v35i5.16542

[17] Xiaolong Li, Gang Pan, Zhaohui Wu, Guande Qi, Shijian Li, Daqing Zhang, Wangsheng Zhang, and Zonghui Wang. 2012. Prediction of Urban Human Mobility Using Large-Scale Taxi Traces and Its Applications. *Frontiers of Computer Science* 6, 1 (2012), 111–121.

[18] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *International Conference on Learning Representations*. https://openreview.net/forum?id=SJiHXGWAZ

[19] Yijun Lin, Nikhit Mago, Yu Gao, Yaguang Li, Yao-Yi Chiang, Cyrus Shahabi, and José Luis Ambite. 2018. Exploiting spatiotemporal patterns for accurate air quality forecasting using deep learning. In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* (Seattle, Washington) *(SIGSPATIAL '18)*. Association for Computing Machinery, New York, NY, USA, 359–368. https://doi.org/10.1145/3274895.3274907

[20] Ziqian Lin, Jie Feng, Ziyang Lu, Yong Li, and Depeng Jin. 2019. DeepSTN+: Context-Aware Spatial-Temporal Neural Network for Crowd Flow Prediction in Metropolis. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01, 1020–1027. https://doi.org/10.1609/aaai.v33i01.33011020

[21] Zhihui Lin, Maomao Li, Zhuobin Zheng, Yangyang Cheng, and Chun Yuan. 2020. Self-Attention ConvLSTM for Spatiotemporal Prediction. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 07, 11531–11538. https://doi.org/10.1609/aaai.v34i07.6819

[22] Ilya Loshchilov and Frank Hutter. 2016. SGDR: Stochastic Gradient Descent with Warm Restarts. *arXiv preprint arXiv:1608.03983* (2016).

[23] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. 2016. Understanding the Effective Receptive Field in Deep Convolutional Neural Networks. 29 (2016). https://proceedings.neurips.cc/paper_files/paper/2016/file/c8067ad1937f728f51288b3eb986afaa-Paper.pdf

[24] Luis Moreira-Matias, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas. 2013. Predicting Taxi-Passenger Demand Using Streaming Data. *IEEE Transactions on Intelligent Transportation Systems* 14, 3 (2013), 1393–1402.

[25] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 1*. 802–810.

[26] Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. 2020. Spatial-Temporal Synchronous Graph Convolutional Networks: A New Framework for Spatial-Temporal Network Data Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 01 (Apr. 2020), 914–921. https://doi.org/10.1609/aaai.v34i01.5438

[27] Yanshen Sun, Kaiqun Fu, and Chang-Tien Lu. 2023. DG-Trans: Dual-level Graph Transformer for Spatiotemporal Incident Impact Prediction on Traffic Networks. *arXiv preprint arXiv:2303.12238* (2023).

[28] Yongxin Tong, Yuqiang Chen, Zimu Zhou, Lei Chen, Jie Wang, Qiang Yang, Jieping Ye, and Weifeng Lv. 2017. The Simpler The Better: A Unified Approach to Predicting Original Taxi Demands based on Large-Scale Online Platforms. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Halifax, NS, Canada) *(KDD '17)*. Association for Computing Machinery, New York, NY, USA, 1653–1662. https://doi.org/10.1145/3097983.3098018

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. 30 (2017). https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[30] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. (2018). https://openreview.net/forum?id=rJXMpikCZ

[31] Pengyang Wang, Kunpeng Liu, Dongjie Wang, and Yanjie Fu. 2021. Measuring Urban Vibrancy of Residential Communities Using Big Crowdsourced Geotagged Data. *Frontiers in Big Data* 4 (2021), 690970. https://doi.org/10.3389/fdata.2021.690970

[32] Pengfei Wang, Daniel Wang, Kunpeng Liu, Dongjie Wang, Yuanchun Zhou, Leilei Sun, and Yanjie Fu. 2023. Hierarchical Reinforced Urban Planning: Jointly Steering Region and Block Configurations. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*. SIAM, 343–351. https://doi.org/10.1137/1.9781611977653.ch39

[33] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu. 2017. PredRNN: Recurrent Neural Networks for Predictive Learning using Spatiotemporal LSTMs. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2017/file/e5f6ad6ce374177eef023bf5d0c018b6-Paper.pdf

[34] Yunbo Wang, Haixu Wu, Jianjin Zhang, Zhifeng Gao, Jianmin Wang, Philip S. Yu, and Mingsheng Long. 2023. PredRNN: A Recurrent Neural Network for Spatiotemporal Predictive Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 2 (2023), 2208–2225. https://doi.org/10.1109/TPAMI.2022.3165153

[35] Fei Wu, Hongjian Wang, and Zhenhui Li. 2016. Interpreting traffic dynamics using ubiquitous urban data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* (Burlingame, California) *(SIGSPACIAL '16)*. Association for Computing Machinery, New York, NY, USA, Article 69, 4 pages. https://doi.org/10.1145/2996913.2996962

[36] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. *arXiv preprint arXiv:1906.00121* (2019).

[37] Mingxing Xu, Wenrui Dai, Chunmiao Liu, Xing Gao, Weiyao Lin, Guo-Jun Qi, and Hongkai Xiong. 2020. Spatial-temporal transformer networks for traffic flow forecasting. *arXiv preprint arXiv:2001.02908* (2020).

[38] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. 2019. Revisiting spatial-temporal similarity: a deep learning framework for traffic prediction. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence* (Honolulu, Hawaii, USA) *(AAAI'19/IAAI'19/EAAI'19)*. AAAI Press, Article 695, 8 pages. https://doi.org/10.1609/aaai.v33i01.33015668

[39] Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, Didi Chuxing, and Zhenhui Li. 2018. Deep multi-view spatial-temporal network for taxi demand prediction. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence* (New Orleans, Louisiana, USA) *(AAAI'18/IAAI'18/EAAI'18)*. AAAI Press, Article 316, 8 pages.

[40] Junchen Ye, Leilei Sun, Bowen Du, Yanjie Fu, Xinran Tong, and Hui Xiong. 2019. Co-Prediction of Multiple Transportation Demands Based on Deep Spatio-Temporal Neural Network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 305–313. https://doi.org/10.1145/3292500.3330887

[41] Xiuwen Yi, Junbo Zhang, Zhaoyuan Wang, Tianrui Li, and Yu Zheng. 2018. Deep Distributed Fusion Network for Air Quality Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 965–973. https://doi.org/10.1145/3219819.3219822

[42] Jiaxuan You, Xiaocheng Li, Melvin Low, David Lobell, and Stefano Ermon. 2017. Deep Gaussian process for crop yield prediction based on remote sensing data. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence* (San Francisco, California, USA) *(AAAI'17)*. AAAI Press, 4559–4565.

[43] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2018. Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence* (Stockholm, Sweden) *(IJCAI'18)*. AAAI Press, 3634–3640.

[44] Fisher Yu and Vladlen Koltun. 2015. Multi-Scale Context Aggregation by Dilated Convolutions. *arXiv preprint arXiv:1511.07122* (2015).

[45] Hao Yuan, Xinning Zhu, Zheng Hu, and Chunhong Zhang. 2020. Deep Multi-View Residual Attention Network for Crowd Flows Prediction. *Neurocomputing* 404 (2020), 198–212. https://doi.org/10.1016/j.neucom.2020.04.124

[46] Zhuoning Yuan, Xun Zhou, and Tianbao Yang. 2018. Hetero-ConvLSTM: A Deep Learning Approach to Traffic Accident Prediction on Heterogeneous Spatio-Temporal Data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 984–992. https://doi.org/10.1145/3219819.3219922

[47] Chi Zhang, Hong-Yu Zhou, Qiang Qiu, Zhichun Jian, Daoye Zhu, Chengqi Cheng, Liesong He, Guoping Liu, Xiang Wen, and Runbo Hu. 2022. Augmented Multi-Component Recurrent Graph Convolutional Network for Traffic Flow Forecasting. *ISPRS International Journal of Geo-Information* 11, 2 (2022), 88.

[48] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence* (, San Francisco, California, USA,) *(AAAI'17)*. AAAI Press, 1655–1661.

[49] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, and Xiuwen Yi. 2016. DNN-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* (Burlingame, California) *(SIGSPACIAL '16)*. Association for Computing Machinery, New York, NY, USA, Article 92, 4 pages. https://doi.org/10.1145/2996913.2997016

[50] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, Xiuwen Yi, and Tianrui Li. 2018. Predicting Citywide Crowd Flows Using Deep Spatio-Temporal Residual Networks. *Artificial Intelligence* 259 (2018), 147–166.

[51] Junbo Zhang, Yu Zheng, Junkai Sun, and Dekang Qi. 2020. Flow Prediction in Spatio-Temporal Networks Based on Multitask Deep Learning. *IEEE Transactions on Knowledge & Data Engineering* 32, 03 (2020), 468–478.

[52] Xiyue Zhang, Chao Huang, Yong Xu, Lianghao Xia, Peng Dai, Liefeng Bo, Junbo Zhang, and Yu Zheng. 2021. Traffic Flow Forecasting with Spatial-Temporal Graph Diffusion Network. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 17 (May 2021), 15008–15015. https://doi.org/10.1609/aaai.v35i17.17761

[53] Ling Zhao, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. 2020. T-GCN: A Temporal Graph Convolutional Network for Traffic Prediction. *IEEE Transactions on Intelligent Transportation Systems* 21, 9 (2020), 3848–3858. https://doi.org/10.1109/TITS.2019.2935152

[54] Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, and Jianzhong Qi. 2020. GMAN: A Graph Multi-Attention Network for Traffic Prediction. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 01 (Apr. 2020), 1234–1241. https://doi.org/10.1609/aaai.v34i01.5477

[55] Jiangchuan Zheng and Lionel M. Ni. 2013. Time-dependent trajectory regression on road networks via multi-task learning. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence* (Bellevue, Washington) *(AAAI'13)*. AAAI Press, 1048–1055.

[56] Yu Zheng, Furui Liu, and Hsun-Ping Hsieh. 2013. U-Air: when urban air quality inference meets big data. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Chicago, Illinois, USA) *(KDD '13)*. Association for Computing Machinery, New York, NY, USA, 1436–1444. https://doi.org/10.1145/2487575.2488188

[57] Yu Zheng, Xiuwen Yi, Ming Li, Ruiyuan Li, Zhangqing Shan, Eric Chang, and Tianrui Li. 2015. Forecasting Fine-Grained Air Quality Based on Big Data. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Sydney, NSW, Australia) *(KDD '15)*. Association for Computing Machinery, New York, NY, USA, 2267–2276. https://doi.org/10.1145/2783258.2788573

[58] Zhengyi Zhou and David S. Matteson. 2015. Predicting Ambulance Demand: a Spatio-Temporal Kernel Approach. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Sydney, NSW, Australia) *(KDD '15)*. Association for Computing Machinery, New York, NY, USA, 2297–2303. https://doi.org/10.1145/2783258.2788570

[59] Ali Zonoozi, Jung-Jae Kim, Xiaoli Li, and Gao Cong. 2018. Periodic-CRN: a convolutional recurrent model for crowd density prediction with recurring periodic patterns. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence* (Stockholm, Sweden) *(IJCAI'18)*. AAAI Press, 3732–3738.