Decoupling Long- and Short-Term Patterns in Spatiotemporal Inference

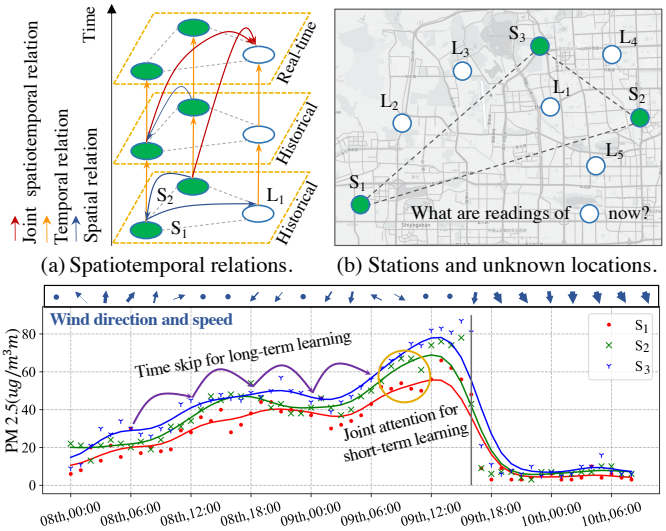
Junfeng Hu, Yuxuan Liang, Zhencheng Fan, Li Liu, Yifang Yin, Roger Zimmermann, *Senior Member, IEEE*

***Abstract*—Sensors are the key to environmental monitoring, which impart benefits to smart cities in many aspects, such as providing real-time air quality information to assist human decision-making. However, it is impractical to deploy massive sensors due to the expensive costs, resulting in sparse data collection. Therefore, how to get fine-grained data measurement has long been a pressing issue. In this paper, we aim to infer values at non-sensor locations based on observations from available sensors (termed spatiotemporal inference), where capturing spatiotemporal relationships among the data plays a critical role. Our investigations reveal two significant insights that have not been explored by previous works. Firstly, data exhibits distinct patterns at both long- and short-term temporal scales, which should be analyzed separately. Secondly, short-term patterns contain more delicate relations including those across spatial and temporal dimensions simultaneously, while long-term patterns involve high-level temporal trends. Based on these observations, we propose to decouple the modeling of short-term and long-term patterns. Specifically, we introduce a joint spatiotemporal graph attention network to learn the relations across space and time for short-term patterns. Furthermore, we propose a graph recurrent network with a time skip strategy to alleviate the gradient vanishing problem and model the long-term dependencies. Experimental results on four public real-world datasets demonstrate that our method effectively captures both long- and short-term relations, achieving state-of-the-art performance against existing methods.**

***Index Terms*—Spatiotemporal Inference, Urban Computing, Graph Neural Network, Attention Mechanism**

I. Introduction

In recent years, numerous sensors have been deployed in different locations to sense the environment. They constantly report spatially-correlated and time-varying readings, such as traffic flows on roads and air quality measurements. Real-time monitoring of spatiotemporal data is of great importance to smart city efforts. For example, air quality information, e.g., the concentration of PM2.5 particles, can support air pollution control and alert the public for health concerns. Unfortunately, one of the critical prerequisites for the above benefits is the fine-grained deployment of sensors, which usually leads to considerable expenditure and high energy consumption [1]. Worse still, even existing sensors might lose readings due to factors such as a poor Internet connection. Thus, how to compensate for the pitfall of lacking sensors has become an urgent and challenging problem. In this paper, we provide one solution by investigating the problem of spatiotemporal inference: given historical and real-time readings of existing sensors, we infer the real-time information *at arbitrary locations* under a graph structure. As exemplified in Figure 1(b), both historical and current readings of S_1–S_3 are leveraged to infer the real-time air quality status of locations L_1–L_5 without actual sensors in those places.



(a) Spatiotemporal relations. (b) Stations and unknown locations.

(c) PM2.5 Readings of S_1, S_2, S_3 and wind information over time.

Fig. 1. (a) Examples of three types of spatiotemporal dependencies. (b) Geographical distribution of stations and unknown locations. (c) PM2.5 readings and wind condition. The curves represent approximated trends and the arrows are time skips for long-term learning. The gray string denotes the time of 16:00, 09/11/2017 where signals do not tally with location relations.

Spatiotemporal inference requires delicate spatial and temporal dependency modeling [1]. Early methods infer nodes based on linear dependencies, such as k-nearest neighbors (KNN) and inverse distance weighting (IDW) [2]. Then, non-linear relations are captured by subsequent approaches such as Gaussian processes (GPs) [3]. However, the Gaussian assumption is rigid and the expensive computation also limits its applicability [4]. Recently, deep learning methods have emerged as a dominant paradigm. Among them, Spatio-Temporal Graph Neural Networks (STGNNs) are widely adopted due to their superior ability to handle non-Euclidean

J. Hu, R. Zimmermann are with the School of Computing, National University of Singapore, Singapore. (email: junfengh@u.nus.edu, rogerz@comp.nus.edu)

Y. Liang is with INTR Thrust, Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China. (email: yuxliang@outlook.com)

L. Liu is with the School of Big Data & Software Engineering, Chognqing University, China. (email: dcsliuli@cqu.edu.cn)

Y. Yin is with Institute for Infocomm Research, A*STAR, Singapore. (email: yin_yifang@i2r.a-star.edu.sg)

Z. Fan is with the Faculty of Engineering and Information Technology, University of Technology Sydney, Australia. (email: Zhencheng.Fan@student.uts.edu.au)

Y. Liang, L. Liu are the corresponding authors of this paper.

sensory data in graph structures [5], [6]. While the spatial relations among locations are naturally defined by the graph [4], the temporal correlations among time points can also be captured, e.g., by concatenating a sequence of readings within a time window as the model input [7]. In other related areas such as forecasting, approaches usually combine GNNs with recurrent neural networks (RNNs) [8] or temporal convolutional networks (TCNs) [9], [10] to learn spatial and temporal dynamics separately. While these methods effectively capture spatiotemporal dependencies, they encounter two major drawbacks when applied to the inference problem.

Firstly, they neglect the difference between long- and short-term patterns in the time series data. For example, readings can fluctuate within a short period in Figure 1(c). On the contrary, when focusing on three curves from 00:00 AM to the next day at 12:00 PM, they still follow an increasing trend. This suggests that long- and short-term relations have inconsistent influences. Unfortunately, the GNN-based inference method [7] concatenates temporal data as features, ignoring this phenomenon. To resolve this, RNN might be feasible as the recurrent structure concentrates more on the latest frames. However, the gradient vanishing problem makes it hard to capture either long-term trends or delicate short-term patterns. Although different strategies, such as adopting adversarial training [11], are proposed to mitigate the problem, they are at the cost of computations and training difficulties.

Secondly, existing inference models lack the ability to effectively learn complex and dynamic spatiotemporal relationships. As illustrated in Figure 1(a), readings are affected by both spatial relations in the graph (i.e., blue arrows) and its historical readings in the temporal dimension (i.e., orange arrows). Moreover, there exist more complicated *joint spatiotemporal dependencies* (i.e., red arrows) that are influenced by sensors at different spatial and temporal positions directly. Unfortunately, the above GNN-RNN structures fail to explicitly consider them, which significantly hobbles the model's performance. Song *et al.* [12] attempt to capture the joint dependencies by a temporal-extended static graph structure. However, this static graph definition struggles to grasp the highly dynamic relations. To illustrate this, in Figure 1(b) and (c), S_2 is geographically close to S_3 but around 16:00 of 09/11/2017, PM2.5 of S_2 is close to that of S_1, possibly due to the fickle wind condition. While several models are introduced to model dynamic relations [13]–[15], the majority of them focus on forecasting, and the inference problem is so far an under-explored research area.

To tackle these issues, we propose a Dual Joint SpatioTemporal Network (DualSTN) for real-time spatiotemporal inference based on graph structures. Our DualSTN decouples short- and long-term learning into dual components: a Joint SpatioTemporal Graph Attention neTwork (JST-GAT) and a Skip Graph Gated Recurrent Unit (SG-GRU). The first component adopts attention blocks to measure the impact between a node and *its spatial neighboring nodes* within *temporal short-term frames*, as the yellow circle shows in Figure 1(c). In this way, JST-GAT learns joint spatiotemporal relations explicitly, discarding the separate learning structures. Meanwhile, impacts are measured by real-time sensor signals, which improves the method's ability to model potential dynamic relations. Inspired by Lai *et al.* [16], the second component consists of a graph GRU with a time skip strategy, aiming to reach the same time span with fewer recurrent steps. This enables the model to capture the long-term temporal trends while ignoring dedicated short-term patterns, as the purple arrows illustrated in Figure 1(c). For long-term dynamic relations, an intuitive way is to learn an adaptive adjacency matrix at each recurrent step as suggested by Wu *et al.* [9] and We *et al.* [17]. However, their transductive design is incompatible with the inductive setting of our task where target locations are not involved during training. Thus, we improve the existing matrix learning method by making node embeddings rely on current input readings. Additionally, we further leverage a graph sampling strategy to train the model [18], which further enhances its generalization ability.

We compare our model with state-of-the-art methods on four real-world datasets. Results show that our DualSTN outperforms the competitors clearly. To evaluate the effectiveness and influence of each module, we also visualize the inference results and the attention weights to interpret DualSTN's ability on modeling long- and short-term patterns as well as dynamic spatiotemporal relations. Our code is available at https://bit.ly/DualSTN and the main contributions are summarized as follows:

- We propose a new framework for spatiotemporal inference, which decouples the long- and short-term pattern learning into separate modules.
- We introduce a JST-GAT module that measures the interactions between nodes in different time and spatial dimensions concurrently, which captures the joint spatiotemporal relations explicitly.
- We propose an SG-GRU to facilitate long-term pattern modeling and optimization, where skip operations are introduced to maintain the same time span with fewer recurrent steps.
- Our DualSTN model achieves state-of-the-art performance on real-world datasets in diverse applications. These results demonstrate the effectiveness and generalization ability of our method.

II. Related Work

A. Spatiotemporal Inference

Spatiotemporal inference aims to infer signals of target locations with surrounding observed readings in a spatiotemporal domain. To solve the task, early statistical methods leverage linear relations modeling. For instance, k-nearest neighbors search and averages neighbor readings as the results while inverse distance weighting (IDW) [2] further utilizes inverse distances as the weights. In addition, several approaches attempt to capture non-linear dependencies, and Kriging [3], [19] is one of the prevalent methods. Based on Gaussian Processes, it designs specialized kernels for the estimation of covariance between nodes and then infers targets by its posterior. However, the Gaussian assumption may not be followed by datasets and in this case, a transformation of non-Gaussian data is required [20]. Wallin *et al.* [21] attempt to map data

TABLE I
MAJOR CHARACTERISTICS OF MODELS. # MEANS THE NUMBER OF.

Model	Year	Temporal Relation	Spatial Relation		Learning Approach
		Method	Method	#-hop	
KNN	None	None	Linear	1-hop	None
IDW [2]	2008	None	Linear	1-hop	None
OKriging [19]	2015	None	Gaussian	1-hop	None
GLTL [24]	2014	Low-Rank Assumption	Low-Rank Assumption	1-hop	Transductive
KCN [4]	2020	None	GNN	1-hop	Inductive
IGNNK [7]	2021	GNN	GNN	n-hop	Inductive
SATCN [27]	2021	TCN	Graph Aggregation	n-hop	Inductive
DualSTN (ours)	New	Joint Attention, GRU	Joint Attention	n-hop	Inductive

into a geo-statistical configuration to weaken the assumption. Besides spatial relations, temporal dependencies are also taken into account. As an example, Yi *et al.* [22] model them by hybrid variables derived from local and global spatiotemporal views. Alternatively, the problem can be regarded as anomaly detection [23] or matrix/tensor completion [24]. Ozkan *et al.* [23] propose an anomaly detection algorithm that adopts a posteriori estimator to fill the missing data while Yu *et al.* [25] complete the matrix by a low-rank matrix assumption.

Recently, deep learning methods have merged as a rife paradigm thanks to their abilities on learning spatiotemporal relations in a data-driven way [26]. Appleby *et al.* [4] propose a Kriging Convolutional Network (KCN) for spatial data inference, which adopts graph convolutional networks to extract dependencies from one-hop neighboring sensors. IGNNK [7] concatenates readings of a sequence along the channel dimension as the inputs of the model to further capture temporal relations. This design, however, treats temporal dependencies uniformly, ignoring inconsistency in the temporal relations. Recently, Wu *et al.* [27] propose SATCN consisting of a Spatial Aggregation Network with multiple aggregation functions to gather diverse spatial information. Then, TCNs are used to capture temporal dependencies. Some solutions concentrate on specific applications. For instance, Cheng *et al.* [28] describe a neural attention model using external features such as weather and POI, named ADAIN. Han *et al.* [29] introduce a novel multi-channel attention model (MCAM) that views external information as feature channels and utilizes LSTM for temporal modeling. However, these deep learning models learn spatial and temporal dependencies separately and external information is not always available, which limits the models' applications.

B. Spatiotemporal Graph Neural Network

Spatiotemporal graph neural networks (STGNNs) are popular for spatiotemporal data modeling nowadays, following mainly two categories. They either couple GNNs with RNNs [30]–[33] or TCNs [9], [34]–[36]. In the first category, GNNs are employed for capturing spatial dependencies, while RNNs are used to model temporal dynamics. For example, Li *et al.* [37] first propose a diffusion convolution that learns the spatial relations through bidirectional random walks on a graph and then capture temporal relations by RNNs. More advanced RNN models such as LSTM and GRU are also utilized in [32], [38]. Xu *et al.* [32] aggregate representations from node neighborhoods as the inputs of a graph GRU while Lai *et al.* [16] use LSTM for long- and short-term modeling, which has a similar motivation to us but only focus on temporal relations. In the second category, TCNs are adopted to learn temporal relationships and enjoy faster running speed than RNNs. For example, Liu *et al.* [34] use the structure to identify more critical data and model it by the proposed adversarial algorithm. Wu *et al.* [9] propose a dilated inception temporal convolution to discover relations with different temporal scales.

In addition, attention mechanisms can be utilized to enhance the performance of STGNNS [38]–[41]. Zheng *et al.* [42] propose a multi-attention network to model spatial and temporal relations independently by attention. Wang *et al.* [43] propose a multi-hop graph attention to calculate weights of context information from multi-hop neighbors. Cai *et al.* [44] explore data periodicity by dividing data into segments. The extracted segments are then fed into the attention network to capture temporal dependencies. Huang *et al.* [45] combine GNNs and attention networks as a spatial gated block and adopt gated linear units (GLU) for temporal dimension, which achieved compelling performances for both short- and long-term forecasting tasks. These designs, however, fail to capture the joint spatiotemporal relations that we aim to address.

To model hidden relations that the adjacency matrix cannot reflect, Li *et al.* [46] and Bai *et al.* [47] propose graph generation methods to learn an adaptive adjacency matrix and STGNNs take these two matrices to learn spatial relations. Unfortunately, the static structure cannot capture dynamic relations and their transductive learning approach is not suitable for our task. To solve the challenge, Shin *et al.* [48] progressively optimize the learned graph for new nodes that are not involved in training, based on their available readings. However, as readings of target locations are completely missing in our problem, this approach is also not applicable.

C. Comparison to Existing Approaches

We compare our model with other inference methods to highlight the differences in this section. KNN, IDW [2], and OKriging [19] are statistical methods, while our DualSTN is a data-driven method. Meanwhile, OKriging is a geo-location method only applied to geographic data. On the contrary, our method is suitable for various datasets. The

transductive GLTL [24] is based on the low-rank assumption, while our model does not require any explicit assumption and is inductive. For the deep learning methods, KCN and KCN-SAGE [4] are one-hop models. Instead, our approach is an n-hop method and also takes temporal dependencies into consideration. IGNNK [7] adopts GNNs but ignores different long- and short-term patterns. SATCN [27] utilizes TCNs to capture temporal relations while our method decouples long- short-term learning and could model spatiotemporal dependencies simultaneously. Table I summarizes the model characteristics.

III. Preliminaries

A. Problem Formulation

In this work, we focus on the real-time spatiotemporal data inference task under a graph structure (see Figure 1 (a), (b)). A graph is represented by $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$, where $\mathcal{V}$ is the node set, $\mathcal{E}$ is the set of edges and $\mathbf{A}$ is the pre-defined adjacency matrix. Suppose we have N_o stations with observed spatiotemporal signals, we denote sensor signals at time t as $\mathbf{X}_t = [\mathbf{x}_t^1, .., \mathbf{x}_t^{N_o}] \in \mathbb{R}^{N_o \times D}$, where D is the number of attributes in a node. The goal aims to use available station readings $[\mathbf{X}_{\tau-T+1}, \mathbf{X}_{\tau-T+2}, .., \mathbf{X}_\tau]$ of time window T to infer signals $\mathbf{Y}_\tau$ of N_u locations at time τ given their spatial relations in the graph $\mathcal{G}$,

$$[\mathbf{X}_{\tau-T+1}, \mathbf{X}_{\tau-T+2}, .., \mathbf{X}_\tau, \mathcal{G}] \xrightarrow{f_\Lambda(\cdot)} [\mathbf{Y}_\tau], \tag{1}$$

where $f_\Lambda(\cdot)$ is the learned mapping function with parameters Λ and assume $N = N_o + N_u$. Note that at any time τ, we only use *historical* and *current* station readings $\mathbf{X}_{\tau-T+1:\tau}$ to infer target locations $\mathbf{Y}_\tau$, which follows the definition of real-time inference as no future readings are available. Further, it is possible that several stations lose readings due to a bad Internet connection or some sensors may be removed or added to the graph. This requires our model to be inductive to various numbers of stations and target locations by design.

B. Graph Convolution Layer

As an essential operation to learn interactions among nodes defined by a graph structure [49], the graph convolution aggregates node features from its neighbors to learn spatial correlations. By stacking convolution layers, the model is capable of learning dependencies from multi-hop neighbors to improve the modeling ability. From the spatial perspective, a graph convolutional layer is formulated as:

$$\mathbf{Z} = \phi(\mathbf{PXW}), \tag{2}$$

where $\mathbf{P} = \mathbf{D}^{-1}(\mathbf{A} + \mathbf{I}) \in \mathbb{R}^{N \times N}$ denotes the normalized adjacency matrix with self-loops, $\mathbf{D}$ is the degree matrix, $\mathbf{X} \in \mathbb{R}^{N \times D}$ are the input readings, $\mathbf{W} \in \mathbb{R}^{D \times F}$ are learnable parameters, $\phi(\cdot)$ is an activation function. Li *et al.* [37] further introduce a diffusion convolution that propagates graph features with K steps:

$$\mathbf{Z} = \phi(\sum_{k=1}^{K} \mathbf{P}^k \mathbf{X} \mathbf{W}_k). \tag{3}$$

To capture hidden graph structures that the pre-defined adjacency matrix cannot reflect, Wu *et al.* [17] propose an adaptive adjacency matrix learning method for the graph convolution, which results in:

$$\mathbf{Z} = \phi(\sum_{k=1}^{K} \mathbf{P}^k \mathbf{X} \mathbf{W}_k + \hat{\mathbf{A}}^k \mathbf{X} \mathbf{U}_k), \tag{4}$$

where $\hat{\mathbf{A}}$ is the adaptive adjacency matrix learned by a network. We term these two graph convolutions as $\Theta_{\star\mathcal{G}}(\mathbf{X}, \mathbf{A})$ and $\Theta_{\star\mathcal{G}}(\mathbf{X}, \mathbf{A}, \hat{\mathbf{A}})$, where Θ are learnable parameters. The important notations in the paper are reported in Table II.

TABLE II
Import notations for data inference.

Notation	Description
N, N_o, N_u	number of nodes / observed sensors / target locations
T	time length for inference
t_s, t_k	short-term time window, number of skip steps
$\mathbf{A}$, $\hat{\mathbf{A}}$	pre-defined / learned adaptive adjacency matrix
$\mathbf{x}_t^i$, $\mathbf{X}_t$, $\mathcal{X}$	readings of the i-th sensor / all sensors at time t / all sensors
$\hat{\mathbf{Y}}_\tau^s$, $\hat{\mathbf{Y}}_\tau^l$	inferred signals of short- / long-term learning at target time τ
$\Theta_{\star\mathcal{G}}(\mathbf{X}, \mathbf{A})$	graph convolution only with pre-defined adjacency matrix
$\Theta_{\star\mathcal{G}}(\mathbf{X}, \mathbf{A}, \hat{\mathbf{A}})$	that with learned adaptive adjacency matrix

IV. Methodology

A. Overview of DualSTN

Figure 2 illustrates the overall framework of our Dual SpatioTemporal Network (DualSTN) which consists of two backbone components for long- and short-term spatiotemporal patterns learning. The short-term Joint SpatioTemporal Graph Attention Network (JST-GAT) first generates *pseudo nodes* for unknown locations for the following attention blocks. Then, stacked graph convolutions and spatiotemporal attention layers are used to learn joint spatiotemporal dependencies, followed by a fully-connected layer to generate short-term inference results. At each recurrent step, the long-term Skip Graph Gated Recurrent Unit (SG-GRU) first learns an adaptive adjacency matrix for the graph in an inductive approach. Then, the graph GRU further takes the learned matrix and hidden states to encode current readings. Last, it integrates short-term results as the input to generate long-term inference results. In the below sections, we first introduce the graph sampling strategy for inductive learning and then describe the details of DualSTN.

B. Graph Sampling for Inductive Learning

In a real-world environment, new sensors might be added to the graph and even existing sensors could retire after some time. In this situation, models need to be compatible with different graphs and input sensors. In addition, they ought to have a better generalization ability, which makes the task more challenging. Previous model Kriging Convolutional Network [4] designed an inductive model but trained the model using a static graph, which is suboptimal and prone to overfitting. Nowadays, approaches solve this by either learning node embedding functions [6] or sampling subgraphs during training [7], [18]. In this paper, we adopt the sampling method which does not involve more parameters. As Figure 3 shows,

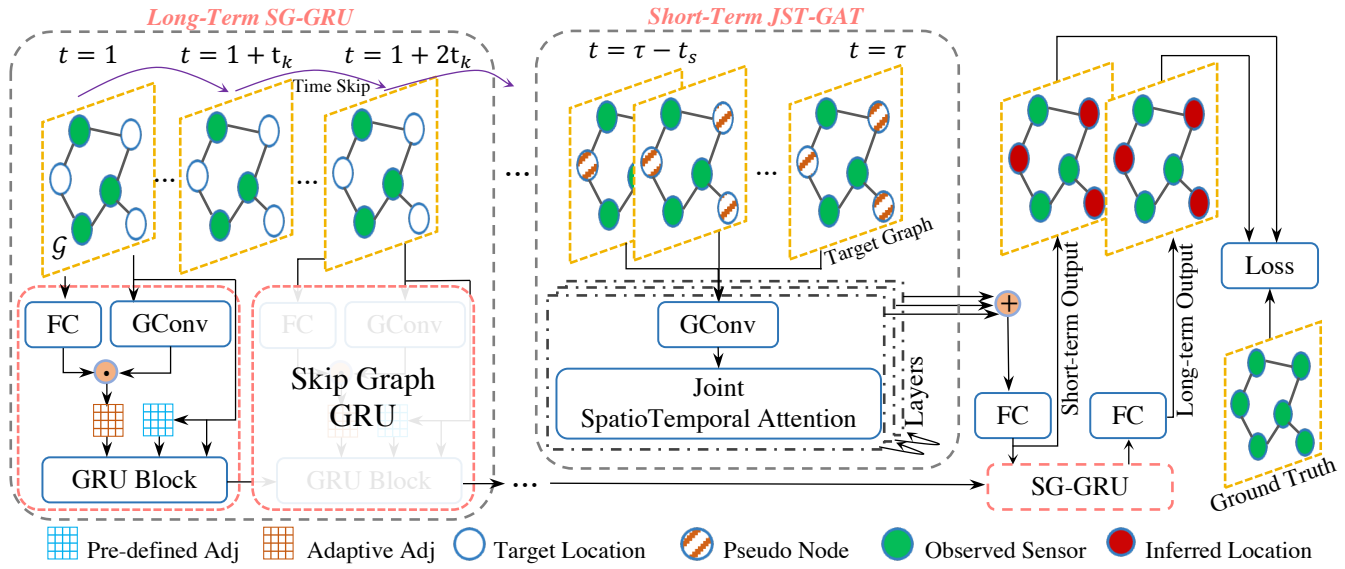


Fig. 2. The framework of the proposed DualSTN model which contains two components: SG-GRU for long-term learning and JST-GAT for short-term learning. Adj means adjacency matrix.

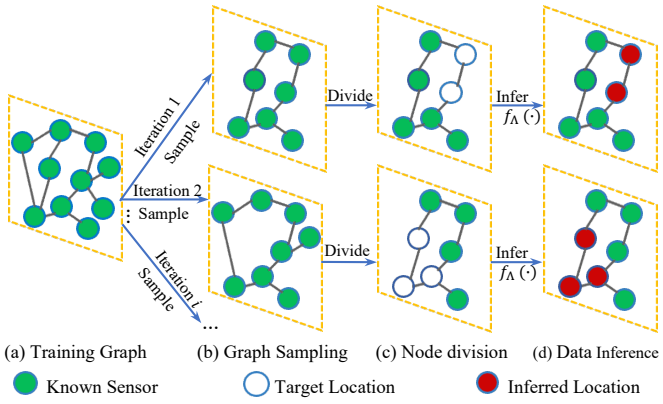


Fig. 3. Illustrations of graph sampling. For each iteration, we sample a subgraph and divide nodes into observed sensors and unknown locations randomly.

given a training graph in (a), for each iteration, we first randomly sample a subgraph in (b). Then in (c), the subgraph is randomly divided into two groups dubbed observed sensors and target locations. In (d), we leverage observed sensors to infer signals of target locations and optimize the network. In this way, the model is less likely to be optimized according to knowledge from the absolute node locations and can capture universal spatiotemporal relations shared among sensors, strengthening its generalization ability. Algorithm 1 describes the graph sampling process in detail. After obtaining the subgraph, we can fetch a batch of data as the inputs for training. Note that we use N to denote the number of nodes in the subgraph in the following sections.

C. Joint Spatiotemporal Graph Attention Network

1) K-nearest Inverse Distance Weighting: As we do not have signals of target locations in a graph, the attention mechanism is suboptimal to be applied directly. Thus, we first

Algorithm 1 Graph Sampling for Each Iteration

Input: graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$, sensor readings $\mathcal{X}$, where $\mathcal{X} \in \mathbb{R}^{T \times N \times D}$.

Output: subgraph $\mathcal{G}_s$, sampled known sensor readings $\mathcal{X}_s$, sampled readings for inference $\mathcal{Y}_s$.

1: Randomly generate two integers N_o, N_u indicating the number of known sensors and unknown locations such that $N_o + N_u \leq N$.
2: Sample a subset of nodes $\mathcal{V}_s$ from $\mathcal{V}$ such that $|\mathcal{V}_s| = N_o + N_u$
3: Divide $\mathcal{V}_s$ into two subsets $\mathcal{V}_k$ and $\mathcal{V}_u$ such that $|\mathcal{V}_k| = N_o$, $|\mathcal{V}_u| = N_u$ and $\mathcal{V}_k \cap \mathcal{V}_u = \emptyset$.
4: Retrieve sensor readings from two subsets: $\mathcal{V}_k \to \mathcal{X}_s$ and $\mathcal{V}_u \to \mathcal{Y}_s$.
5: Retrieve edges $\mathcal{E}_s$ from $\mathcal{V}_s$.
6: Construct the adjacency matrix $\mathbf{A}_s$ using $\mathcal{E}_s$ and $\mathbf{A}$.
7: **return** $\mathcal{G}_s$, $\mathcal{X}_s$, $\mathcal{Y}_s$.

calculate initial readings for them by k-nearest inverse distance weighting (k-IDW). To be specific, we fill short-term values of the locations (termed *pseudo nodes*) at time τ and its short-term neighbor frames in the window $[\tau - t_s, \tau - 1]$. As shown in the yellow arrows of Figure 4, k-IDW follows the idea of k-nearest neighbors that first searches the spatially k-nearest observed sensors for each target location. Then, the inverse distances from the location to its neighbors are utilized as weights to calculate the mean:

$$\tilde{\mathbf{x}}_{t,i} = \frac{\sum_{j=1}^{k} \mathbf{x}_{t,j} \odot {d_{i,j}}^{-\rho}}{\sum_{j=1}^{k} {d_{i,j}}^{-\rho}}, \tag{5}$$

where $t \in [\tau - t_s, \tau]$, k denotes the assigned number of nearest neighbors, $d_{i,j}$ is the distance between a tarrget location i and the neighbor j, ρ means the decay rate, and $\odot$ represents the Hadamard product. After obtaining pseudo nodes, they

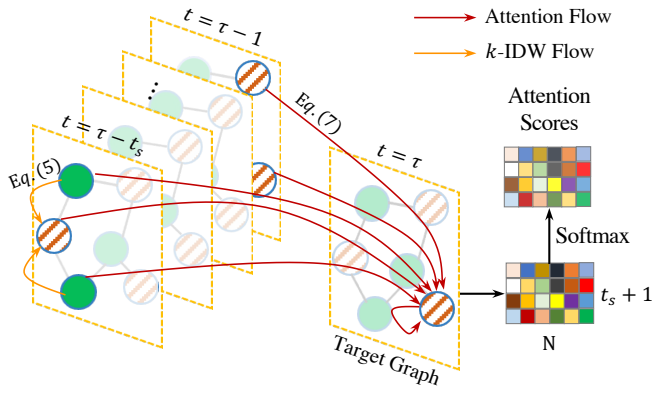


Fig. 4. Illustrations and information flows of k-nearest inverse distance weighting and the joint spatiotemporal attention. Only a limited number of flows for one node are drawn.

can be used to compute attention scores and we regard pseudo nodes and observed sensors uniformly, notated as $\mathbf{X}_t = [\mathbf{x}_{t,1}, .., \mathbf{x}_{t,N_o}, \tilde{\mathbf{x}}_{t,1}, .., \tilde{\mathbf{x}}_{t,N_u}] \in \mathbb{R}^{N \times D}$ below.

2) Joint Spatiotemporal Attention: Spatiotemporal data are influenced by conditions from surrounding locations which are highly interactive and difficult to capture the patterns. To learn these relationships, attention networks are proven to be an effective tool [39]. Among them, researchers mainly utilize individual attention blocks to handle the spatial relations and temporal dynamics respectively, followed by a fusion module to integrate them [41], [42], [50]. However, these models fail to explicitly consider joint spatiotemporal dependencies directly. For instance, assuming a north wind, PM2.5 particles will be blown toward the south over time. In addition, a car accident will clog traffic on an upstream road after a short time. The phenomena involve synchronous spatiotemporal shifts and are hard for separate attention modules to model. We propose a joint spatiotemporal attention mechanism to capture them simultaneously, as illustrated in Figure 4.

Given features of a target frame $\mathbf{Z}_\tau^{l-1}$ at layer $l-1$ and its neighbor frames $\mathbf{Z}_t^{l-1}$ in the short-term window $[\tau - t_s, \tau - 1]$, we first employ a graph convolution layer with skip connection to learn the spatial relations in each frame:

$$\mathbf{Z}_t^l = \gamma \mathbf{Z}_t^{l-1} + \mu \Theta_{\star\mathcal{G}}^l \left(\mathbf{Z}_t^{l-1}, \mathbf{A}\right) \quad t \in [\tau - t_s, \tau], \tag{6}$$

where $\mathbf{Z}_t^0 = \mathbf{X}_t$, γ and μ are hyperparameters for skip connection. Here, the adjacency matrix $\mathbf{A}$ defines static spatial relations as a prior so the attention mechanism relieves from learning it again, reducing learning difficulties. Then, the attention operation captures joint spatiotemporal dependencies. We first calculate attention weights between sensor $\mathbf{z}_{\tau,i}^l$ and other sensors $\mathbf{z}_{t,j}^l$ in the window formulated by:

$$e_{t,i,j}^l = \mathbf{v_a}^\top \tanh(\mathbf{W_a} \mathbf{z}_{\tau,i}^l + \mathbf{U_a} \mathbf{z}_{t,j}^l + \mathbf{b_a}), \tag{7}$$

$$e_{t,i,j}^l = \frac{\exp(e_{t,i,j}^l)}{\sum_{t=\tau-t_s}^{\tau} \sum_{j=1}^{N} \exp(e_{t,i,j}^l)}, \tag{8}$$

where $\mathbf{v_a}, \mathbf{b_a} \in \mathbb{R}^F$, $\mathbf{W_a}, \mathbf{U_a} \in \mathbb{R}^{D \times F}$ are learnable parameters. Note that the parameters are shared across all frames to reduce the computational cost. We also compute attention scores within the target frame and in this situation, the block degrades into spatial attention. Finally, we obtain an attention map $\mathbf{E}^l \in \mathbb{R}^{(t_s+1) \times N \times N}$, where the second dimension N refers to sensor features at the target time τ and the third dimension N denotes sensors in the $t_s + 1$ frames. Next, we use features of short-term frames and the attention map to learn short-term inference representations:

$$\mathbf{z}_{\tau,i}^l = \sum_{t=\tau-t_s}^{\tau} \sum_{j=1}^{N} \mathbf{E}_{t,i,j}^l \mathbf{z}_{t,j}^l. \tag{9}$$

We stack the joint attention blocks for L layers. On the top layer L, a fully connected layer is used to generate the short-term outputs:

$$\hat{\mathbf{Y}}_\tau^s = \mathbf{Z}_\tau^L \mathbf{W}_{fs} + \mathbf{b}_{fs}, \tag{10}$$

where $\mathbf{W}_{fs} \in \mathbb{R}^{F \times D}$, $\mathbf{b}_{fs} \in \mathbb{R}^D$ are parameters and $\hat{\mathbf{Y}}_T^s \in \mathbb{R}^{N \times D}$ are short-term inference results. Finally, the joint spatiotemporal dependencies can be learned by a single JST-GAT without separate modules. Moreover, the attention block aids interpretability by visualizing weights to understand how the model learns the spatiotemporal relations.

D. Skip Graph Gated Recurrent Unit

1) Inductive Dynamic Graph Generation: To model the hidden relations among nodes, previous works learn an adaptive adjacency matrix during training and it remains static during testing [9], [17], [47]. However, this disregards the dynamic dependencies of the graph structure over the timeline. Later, Li *et al.* [46] model the dynamic connections by learning an adaptive matrix at each step of a recurrent network. Unfortunately, the method significantly relies on embeddings of training nodes which is not available in the inductive setting. To solve these challenges, we propose an inductive graph generation module that updates the adjacency matrix in an inductive fashion based on [9]. To be specific, at time step t, we use the historical hidden states $\mathbf{H}_{t-t_k}$, $\mathbf{X}_t$ and $\mathbf{A}$ to learn graph structure information. Here, the t_k is a skip step described below. The node embedding is replaced by a fully-connected layer $FC(\cdot)$ taking $\mathbf{H}_{t-t_k}$ as input. In summary, the adaptive adjacency matrix $\hat{\mathbf{A}}_t$ at time t is calculated by:

$$\begin{aligned}
\mathbf{M}_t^1 &= \tanh(\Theta_{1\star\mathcal{G}}(\mathbf{X}_t, \mathbf{A}) \odot FC_1(\mathbf{H}_{t-t_k})), \\
\mathbf{M}_t^2 &= \tanh(\Theta_{2\star\mathcal{G}}(\mathbf{X}_t, \mathbf{A}) \odot FC_2(\mathbf{H}_{t-t_k})), \\
\hat{\mathbf{A}}_t &= \mathrm{ReLU}(\tanh(\alpha(\mathbf{M}_t^1 \mathbf{M}_t^{2\top} - \mathbf{M}_t^2 \mathbf{M}_t^{1\top}))),
\end{aligned} \tag{11}$$

where $\mathbf{M}_1^t$ and $\mathbf{M}_2^t \in \mathbb{R}^{N \times F}$ are source node encoder and target node encoder, respectively, $\sigma(\cdot)$ is the sigmoid activation function and α is the saturation rate hyperparameter.

2) Skip Graph Gated Recurrent Unit: GRU is one of the recurrent structures designed to capture historical temporal information in a recurrent way [51]. However, the gradient vanishing and overwhelming state estimation of the latest inputs cause difficulties to capture long-term temporal patterns [52]. To alleviate this, motivated by [16] we propose a graph GRU with skips to maintain a temporal span with fewer recurrent steps. To be specific, hidden states are updated using

Algorithm 2 Training Procedure of DualSTN

Input: graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, sensor readings of training set $\mathcal{X}$, time window T; initialized model DualSTN().
Output: optimized learnable weights.

for $i = 1 \rightarrow Num\_Iteration$ **do**
  Initialize batch list $\mathbf{X_b} = []$, $\mathbf{Y_b} = []$.
  {Sampling Graph}
  $\mathcal{G}_s$, $\mathbf{X_s}$, $\mathbf{Y_s}$ = Graph_Sampling($\mathcal{G}$, $\mathcal{X}$).
  for $j = 1 \rightarrow Batch\_Size$ **do**
    Randomly choose a start time t.
    Append $\{\mathbf{X_s}\}_{t:t+T}$ to $\mathbf{X_b}$.
    Append $\{\mathbf{Y_s}\}_{t+T}$ to $\mathbf{Y_b}$.
  end for
end for
$\hat{\mathbf{Y}}^s_{\mathbf{b}}$, $\hat{\mathbf{Y}}^l_{\mathbf{b}} = \mathrm{DualSTN}(\mathcal{G}_s, \mathbf{X_b})$.
Compute $\mathrm{MAE}(\hat{\mathbf{Y}}^s_{\mathbf{b}}, \mathbf{Y_b}) + \mathrm{MAE}(\hat{\mathbf{Y}}^l_{\mathbf{b}}, \mathbf{Y_b})$ and derive the gradients.
Update learnable weights using the optimizer.

historical hidden states of a certain number of skips t_k, which can be formulated as:

$$\begin{aligned}
\mathbf{r}_t &= \sigma(\Theta_{r\star\mathcal{G}}(\mathbf{X}_t|\mathbf{H}_{t-t_k}, \mathbf{A}, \hat{\mathbf{A}}) + \mathbf{b}_r),\\
\mathbf{u}_t &= \sigma(\Theta_{u\star\mathcal{G}}(\mathbf{X}_t|\mathbf{H}_{t-t_k}, \mathbf{A}, \hat{\mathbf{A}}) + \mathbf{b}_u),\\
\mathbf{c}_t &= \tanh(\Theta_{c\star\mathcal{G}}(\mathbf{X}_t|(\mathbf{H}_{t-t_k} \odot \mathbf{r}_t), \mathbf{A}, \hat{\mathbf{A}}) + \mathbf{b}_c),\\
\mathbf{H}_t &= \mathbf{u}_t \odot \mathbf{H}_{t-t_k} + (1 - \mathbf{u}_t) \odot \mathbf{c}_t,
\end{aligned} \tag{12}$$

where | means concatenation, $\mathbf{r}_t$ and $\mathbf{u}_t \in \mathbb{R}^{N\times F}$ are reset gate and update gate, respectively. Through the recurrent skip, the module is encouraged to focus on the high-level long-term temporal patterns, which ignores delicate relations among consecutive frames. The decreased recurrent steps also facilitate the optimization process. At the last recurrent step at which the short-term inference results $\hat{\mathbf{Y}}^s_\tau$ are computed, we feed $\hat{\mathbf{Y}}^s_\tau$ to the graph GRU to compute the corresponding hidden states $\mathbf{H}_\tau$. Finally, a fully connected layer is used to obtain the long-term inference outputs:

$$\hat{\mathbf{Y}}^l_\tau = \mathbf{H}_\tau \mathbf{W}_{fl} + \mathbf{b}_{fl}, \tag{13}$$

where $\mathbf{W}_{fl} \in \mathbb{R}^{F\times D}$, $\mathbf{b}_{fl} \in \mathbb{R}^D$ are parameters, and $\hat{\mathbf{Y}}^l_\tau \in \mathbb{R}^{N\times D}$ are long-term inference outputs.

E. Loss Function and Training Procedure

To strengthen the generalization ability, we train our model by reconstructing all sensor signals instead of just target locations like Appleby *et al.* [4] and simultaneously optimize the MAE loss of short-term outputs $\hat{\mathbf{Y}}^s_\tau$ as well as long-term results $\hat{\mathbf{Y}}^l_\tau$:

$$\mathcal{L} = \frac{1}{N}\left|\mathbf{Y}_\tau - \hat{\mathbf{Y}}^l_\tau\right| + \frac{1}{N}\left|\mathbf{Y}_\tau - \hat{\mathbf{Y}}^s_\tau\right|. \tag{14}$$

The training procedure of DualSTN can be summarized in Algorithm 2. For each iteration, we randomly sample a subgraph and its corresponding sensor readings to train the model. Note that we adopt the same subgraph and node division for each batch to simplify the implementation.

TABLE III
DATASET STATISTICS. #: THE NUMBER. TRAFFIC SPD: TRAFFIC SPEED. RN-DIS/GEO-DIS: ROAD-NETWORK/GEOSPATIAL DISTANCE.

Dataset	METR-LA	PeMS-Bay	NREL	BJ-Air
Category	Traffic Spd	Traffic Spd	Solar Energy	Air Quality
Adjacency Matrix	Rn-Dis	Rn-Dis	Geo-Dis	Geo-Dis
# Sensors	207	325	137	35
# Time points	34,272	52,116	105,120	10,228
Frequency	5-min	5-min	5-min	1-hour
Mean	58.45	62.62	15.96	60.99
Standard Deviation	13.08	9.58	9.86	65.31

V. EXPERIMENTS

A. Experimental Settings

1) Datasets: We evaluate the performances of DualSTN on four real-world spatiotemporal datasets in diverse application scenarios: 1) **METR-LA**[1] [37]: traffic speed dataset collected from 207 sensors in the highway of Los Angeles from 01/05/2012 to 30/06/2012. 2) **PeMS-Bay**[2] [37]: a traffic speed dataset collected by California Transportation Agencies containing 325 sensors in the Bay Area from 01/01/2017 to 13/05/2017. 3) **NREL**[3] [53]: energy datasets provided by the National Renewable Energy Laboratory and we choose a subset of Alabama Solar Power Data. The dataset contains 137 photovoltaic power plant readings collected in 2006. 4) **BJ-Air**[4]: air quality index dataset from 35 air quality stations in Beijing and we consider the PM2.5 observations.

We construct the pre-defined adjacency matrix $\mathbf{A}$ based on either road network distance or geospatial distance $dist(v_i, v_j)$. The road network distance is available in the dataset and we compute geospatial distance using Haversine formula, given the longitude and latitude:

$$\begin{aligned}
dist(v_i, v_j) = 2r \arcsin\Bigg(\sin^2\left(\frac{\varphi_j - \varphi_i}{2}\right) + \\
\cos(\varphi_i)\cos(\varphi_j)\sin^2\left(\frac{\lambda_j - \lambda_i}{2}\right)\Bigg)^{\frac{1}{2}},
\end{aligned} \tag{15}$$

where $r = 6371$ is the radius of the earth, (φ_i, λ_i) means the longitude and latitude of the sensor v_i. Then the Gaussian kernel method [54] is applied:

$$\mathbf{A}_{i,j} = \exp\left(-\frac{1}{2}\frac{dist(v_i, v_j)^2}{\sigma^2}\right), \tag{16}$$

where σ is the standard deviation. In the case of the directed graph that contains bi-directional adjacency matrices $\mathbf{A}_f$ and $\mathbf{A}_b$, we use the same network to model them which is equivalent to $\mathbf{A} = (\mathbf{A}_f + \mathbf{A}_b)/2$. We summarize the statistics of datasets in Table III.

2) Baseline Methods: We compare the performances of our model with 7 baselines as follows:

- **KNN**: K-nearest neighbors interpolate readings of unknown locations by averaging the k-nearest sensors in the spatial dimension.

[1]https://github.com/liyaguang/DCRNN
[2]https://github.com/liyaguang/DCRNN
[3]https://www.nrel.gov/grid/solar-power-data.html
[4]https://www.biendata.xyz/competition/kdd_2018/

- **IDW**: Inverse distance weighting utilizes distances between nodes to calculate a weighted average of available nodes for each unknown location [2].
- **OKriging**: Ordinary kriging is a classic statistical interpolation method based on the geospatial locations of the sensors and Gaussian processes [19]. We evaluate the performance of OKriging using the package PyKrige[5]. Note that OKriging is not applicable for road network distance, so we just report the performances on NREL and BJ-Air datasets.
- **GLTL:**[6] Greedy Low-rank Tensor Learning is a low-rank tensor learning framework for spatiotemporal data cokriging and forecasting which handles various properties in the data. Moreover, a fast greedy algorithm is proposed to learn the tensor efficiently.
- **KCN**, **KCN-SAGE**[7]: Kriging Convolutional Network first searches k-nearest neighbors for a target location. Then, it constructs a graph structure for the K + 1 nodes as inputs of GNNs to interpolate signals [4]. KCN-SAGE is a variant based on graph sampling and aggregating [6].
- **IGNNK**[8]: Inductive Graph Neural Network Kriging is a state-of-the-art model trained in an inductive approach [7]. It regards sequential reading as the feature of a GNN to learn spatial and temporal relations.
- **SATCN**[9]: Spatial Aggregation and Temporal Convolution Network contains a spat of spatial aggregators using signals' statistic features for spatial modeling. Meanwhile, TCNs are leveraged to capture temporal dependencies [27].

3) Evaluation Metrics: We utilize three criteria to evaluate models: the root mean square error (RMSE), the mean absolute error (MAE), and the mean absolute percentage error (MAPE). All of them are frequently used in regression problems:

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i\in N}(y^i - \hat{y}^i)^2}, \tag{17}$$

$$\mathrm{MAE} = \frac{1}{N}\sum_{i\in N}\left|y^i - \hat{y}^i\right|, \tag{18}$$

$$\mathrm{MAPE} = \frac{1}{N}\sum_{i\in N}\frac{\left|y^i - \hat{y}^i\right|}{y^i}, \tag{19}$$

where y^i is the ground truth signal of a sensor and $\hat{y}^i$ denotes inferred signals.

4) Implementation Details: Our DualSTN and other deep-learning baselines are implemented with PyTorch 1.7 and trained on a Quadro RTX 6000 GPU. We use a historical time window $T = 25$ to infer the real-time readings in which the time window for short-term learning is $t_s = 3$. The skip step of GRU t_k equals 4, i.e., we feed readings of sensors into the network every 4 steps. For the hyperparameters, we set decay rates ρ and λ to 1, saturation rate α to 2, and impact factors γ, μ to 0.1 and 0.9. The activation function ϕ is ReLU. The number of layers for JST-GAT is 3 and the size of hidden states for graph GRU is 16. All learnable parameters are initialized with the Xavier [55]. The model is trained by the Adam [56] optimizer and the learning rate of 10^{-3}. Note that we keep the same settings for all datasets, verifying the generalization ability of our model. For the dataset division, we use the first 70% of the time frames to train models and validate or test models using the following 20% and 10%, respectively. For the sensor division, we manually leave 50% of the sensors out for testing, dubbed testing sensors, and the remaining 50% as training sensors. At each epoch, we randomly mask 50% of the training sensors as unknown locations. When evaluating models, we use all training sensors to interpolate all testing sensors. To ensure a fair comparison, we use the same division of training and testing sensors and train each deep learning model five times independently to report the average results and the standard deviations.

[5]https://geostat-framework.readthedocs.io/projects/pykrige/en/stable
[6]https://roseyu.com/code.html
[7]https://github.com/tufts-ml/KCN
[8]https://github.com/Kaimaoge/IGNNK
[9]https://github.com/Kaimaoge/SATCN

B. Model Comparison

In this section, we compare the performances of our DualSTN with all baselines, and the results are summarized in Table IV. We observe that statistical methods have worse results than data-driven approaches. This is chiefly because they are designed to capture linear or Gaussian relations, which is suboptimal to measure complex dependencies. For deep learning methods, as they could learn complicated non-linear spatiotemporal dependencies from data, we find that even the spatial method KCN outperforms GLTL which considers both spatial and temporal dependencies. Meanwhile, IGNNK outperforms KCN because IGNNK adopts n-hop GNNs and learns from temporal information. For our model, we observe that it outperforms all baselines on LETR-LA, PeMS-Bay, and NREL and competitive results on the BJ-Air dataset. It results from that DualSTN could learn spatiotemporal relationships simultaneously and the decoupled design makes the model easier to distinguish long- and short-term patterns. In addition, DualSTN also has fewer parameters than other models, which also demonstrates its learning ability. The reason might be that our model uses a single module to learn spatial and temporal relations. Thus, we do not need to stack many layers to enlarge the receptive field, reaching nodes with far distances. For the BJ-Air dataset, both our DualSTN and SATCN achieved commensurate performance and surpass other baselines. SATCN designs special aggregators like the standard deviation aggregator, which might be beneficial for this dataset with a large deviation. However, DualSTN has five times fewer parameters than SATCN. Furthermore, we notice that on the BJ-Air dataset, the results of the three deep learning models have larger variations, which means that the training process is not stable. We conjecture the limited number of sensors and the large standard deviation of readings cause difficulties in learning general spatiotemporal dependencies.

C. Ablation Study

Our model is largely built upon two motivations (i.e., decoupling long- short-term patterns and joint spatiotemporal

TABLE IV
PERFORMANCE COMPARISON AND THE NUMBER OF LEARNABLE PARAMETERS FOR DIFFERENT METHODS ON FOUR DATASETS. K: THOUSAND.

Model	METR-LA			PeMS-Bay			#Params
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	
KNN	7.65±0.00	11.20±0.00	0.183	6.45±0.00	12.16±0.00	0.112	–
IDW	7.78±0.00	11.53±0.00	0.188	6.38±0.00	11.78±0.00	0.103	–
OKriging	–	–	–	–	–	–	–
GLTL	7.71±0.00	11.03±0.00	0.186	5.20±0.00	8.90±0.00	0.0983	–
KCN	7.19±0.04	10.55±0.07	0.183	4.70±0.07	8.12±0.06	0.095	18K
KCN-SAGE	7.06±0.04	10.05±0.07	0.179	4.38±0.06	7.50±0.07	0.092	15K
IGNNK	6.93±0.09	10.59±0.05	0.175	4.06±0.06	7.12±0.08	0.090	45K
SATCN	7.03±0.05	10.51±0.07	0.176	4.04±0.03	6.96±0.02	0.092	66K
DualSTN (ours)	**6.73±0.03**	**9.93±0.02**	**0.171**	**3.95±0.06**	**6.68±0.04**	**0.088**	**12K**
Model	NREL			BJ-Air			#Params
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	
KNN	3.28±0.00	4.62±0.00	1.031	18.35±0.00	29.41±0.00	1.044	–
IDW	3.08±0.00	4.46±0.00	0.932	17.80±0.00	28.21±0.00	0.921	–
OKriging	2.81±0.00	4.23±0.00	0.855	17.98±0.00	28.62±0.00	0.954	–
GLTL	3.20±0.00	4.49±0.00	0.937	16.33±0.00	26.94±0.00	0.933	–
KCN	1.71±0.08	2.90±0.08	0.739	14.63±0.20	25.39±0.32	0.743	18K
KCN-SAGE	1.65±0.05	2.84±0.07	0.701	14.40±0.26	24.98±0.27	0.725	15K
IGNNK	1.53±0.06	2.75±0.08	0.682	13.79±0.22	24.95±0.29	0.651	45K
SATCN	1.69±0.05	2.90±0.02	0.735	**12.93±0.11**	23.39±0.22	0.560	66K
DualSTN (ours)	**1.49±0.03**	**2.69±0.02**	**0.643**	13.01±0.20	**23.30±0.28**	**0.554**	**12K**

learning). A natural question is whether they are effective. In the section, we implement three variants of DualSTN to verify the effectiveness of components described as follows:

- **LongSTN**: This variant removes the short-term learning module and only contains the skip graph gated recurrent unit. The inference results are outputs of the GRU's last recurrent step.
- **ShortSTN**: This variant removes the skip GRU and remains the joint spatiotemporal attention graph network which only takes short-term frames as the inputs to infer unknown locations.
- **DualTCN**: Our model uses attention blocks to learn temporal dependencies in the short-term module and another intuitive opinion is leveraging TCNs. Thus, this variant replaces the attention module with the temporal convolution network (TCN) that follows the same structure as [17] and the long-term module remains the same.

We evaluate the performance of three variants on four datasets and illustrate the results in Figure 5. We find that ShortSTN performs better than LongSTN while both are much worse compared to DualSTN. These observations demonstrate the following conclusions. 1) Sensor readings are more relevant to short-term patterns but still follow the trend of long-term ones. 2) Decoupling long- and short-term learning improves performance as they provide information from different perspectives. For the TCN variant, we observe that DualSTN has a better performance compared to DualTCN and argue that this is because the joint attention module explicitly takes joint spatiotemporal relations into consideration, which reduces the challenge of modeling complex relations.

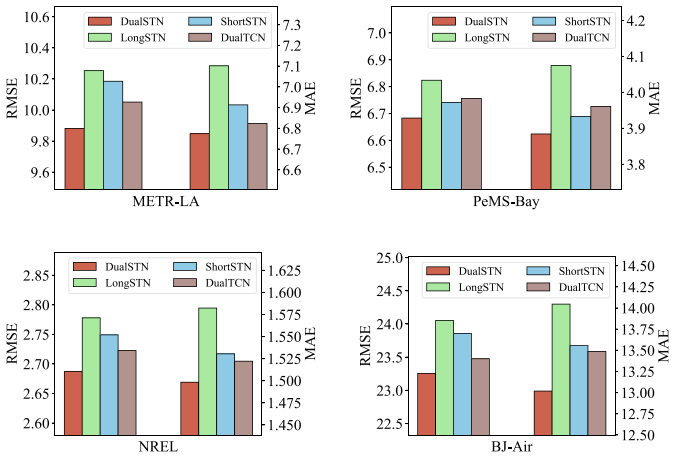


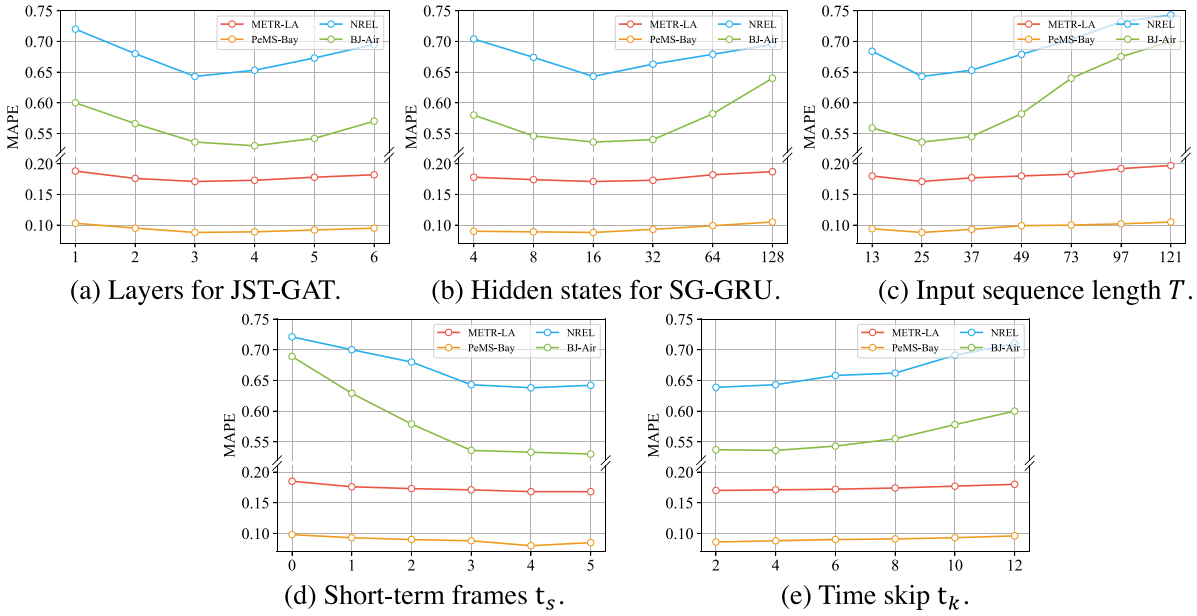
Fig. 5. Ablation studies. The DualSTN consistently achieves the best RMSE and MAE results against other variants.

D. Hyperparameter Study

In this section, we study the performance of our DualSTN under different hyperparameter settings and report MAPE results. In each study, we modify the setting of corresponding hyperparameters and keep others unchanged. All the experiments are conducted on four datasets.

1) Effects of number of layers of JST-GAT: We adjust the number of layers L in the joint spatiotemporal attention module and report the results in Figure 6(a). The model performances first become better and achieve best at 4 layers for the BJ-Air dataset and 3 layers for the rest. Then, MAPE results remain stable or start to increase slightly. According to these observations, we uniformly keep the number of layers as 3 to reduce the computational cost.

2) Effects of size of hidden states of SG-GRU: We change the size of the graph GRU's hidden states from 4 to 128. As shown in Figure 6(b), we discover that the performances



(a) Layers for JST-GAT. (b) Hidden states for SG-GRU. (c) Input sequence length T.

(d) Short-term frames t_s. (e) Time skip t_k.

Fig. 6. Parameters study among different variants on METR-LA, PeMS-Bay, NREL, and BJ-Air datasets.

increase until reaching the size of 16 or 32 for four datasets. Then they start to decrease, indicating that the model tends to overfit. Accordingly, we set the size to 16.

3) Effects of time window T: We keep the skip step in the graph GRU $t_k = 4$ and adjust the input time window T to evaluate GRU's capability of learning long-term patterns. As expected in Figure 6(c), a longer input sequence cannot guarantee better results. Instead, the model crashes over a long time window because of its gradient vanishing problem of GRU on long-term modeling. In this scenario, the encoded long historical features become noises of hidden states, impeding the model performance.

4) Effects of number of short-term frames t_s: Next, we modify the number of short-term frames t_s. As illustrated in Figure 6(d), as t_s increases, the performance first increases fast and then levels off. As a larger t_s uses frames modeled by SG-GRU, the delicate learning module JST-GAT is redundant to capture them again. To this end, we set t_s to 3 and leverage the graph GRU for learning these patterns.

5) Effects of time skip t_k: Finally, we keep $t_s = 3$, $T = 25$ and adjust time skip t_k to evaluate its influence. As shown in Figure 6(e), the results consistently decrease and especially, this decrease speeds up as t_k increases. The reason is that as the time span becomes too large, the temporal dependencies between input frames become sparse that the model cannot capture. Thus, we choose t_k to 4 to ensure no frame overlap between the short-term and long-term modules which also saves running time.

From the study, we observe that DualSTN is able to achieve satisfying performances over all datasets using the same hyperparameter setting. This means that our model is insensitive to different application domains, which relieves the demand for hyperparameter searching and is beneficial to real-world deployment. It could be caused by the fewer learnable parameters and the learning effectiveness of the model.

E. Case Study

1) Long- and short-term patterns inconsistency: To study how our DualSTN captures the long- and short-term patterns, we visualize the short-term results $\hat{\mathbf{Y}}^s$ and the final results $\hat{\mathbf{Y}}^l$ that integrates information of both terms. Figure 7 shows results and ground truth of META-LA from 16:30, 23/06/2023, and the BJ-Air dataset from 14/01/2018, in which we have three observations. 1) In the red boxes, the truth signals fluctuate while having a flat trend. The short-term inferred signals, aligning with the ground truth, also oscillate as the JST-GAT only focuses on short-term patterns. Then by involving trends from long-term patterns, the final outputs become stable. 2) In the blue boxes, signals follow an upward or downward trend. The long-term outputs are accurate compared to the short-term results. This suggests the tendency is important in this scenario and SG-GRU could capture it precisely. 3) In the gray boxes where a sudden change in readings happens, the short-term outputs outperform the long-term results. This is because the historical tendency in SG-GRU does not tally with this sudden change. These discoveries mean that our model handles short- and long-term patterns, and JST-GAT and SG-GRU can contribute to the final results in different aspects.

2) Joint spatiotemporal Dependencies: The attention block in our model provides interpretability by indicating the dependencies between two nodes. As the motivation here is to capture joint spatiotemporal relations, we conduct a case study using the BJ-Air dataset from 0:00 to 12:00 on 03/09/2017 and visualize the attention weights of an inferred location to investigate this ability. For succinctness, we use a center station S_{21} as the target location and compute attention weights between its signals and those of equably sampled 13 sensors at different times. Figure 8(a) illustrates their locations and Figure 8(b) shows the attention scores of 12 consecutive time steps, where stations are drawn according to their relative orientations to S_{21}. Given the southeastern condition, we have the following

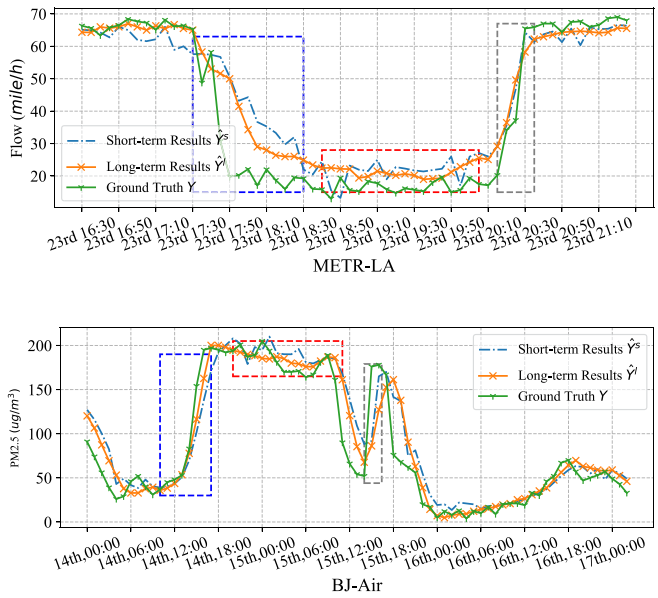
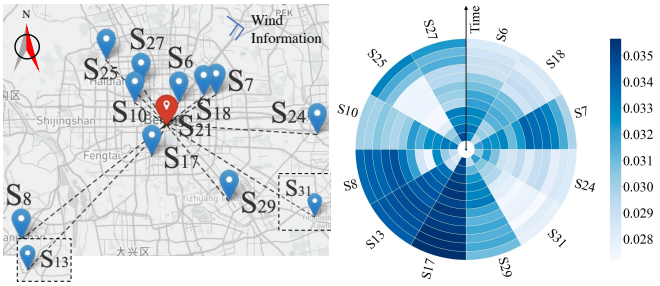


Fig. 7. Visualizations of short-term and final long-term inference results compared to the ground truth on METR-LA and BJ-Air datasets. The time starts from 16:30, 23/06/2023, and 0:00, 14/01/2018, respectively.

conclusions. 1) Stations S_8, S_{13}, and S_{17} have larger weights, meaning that these stations contribute more to the inference of S_{21}. 2) The pre-defined adjacency matrix could reflect the spatial relations regarding geographical information, as near stations have a large influence (e.g., S_{29} and S_{31} to S_{21}). 3) However, it cannot guarantee genuine spatiotemporal dependencies. For instance, S_{13} is excessively far away from S_{21} compared to S_{18} but has an even larger impact. In this situation, the dynamic dependencies become dominant. 4) The weights of S_8 increase over time, mightily due to the change of wind speed. This means that joint spatiotemporal attention is capable of capturing relations across time and space. These observations verify that our model captures both static spatial relations, dynamic implicit dependencies, and joint spatiotemporal relations simultaneously even without knowing possible external factors. This compelling learning ability also provides the possibility of applying the module to other tasks that require delicate spatial and temporal modeling, such as video object segmentation [57], [58].



(a) Air quality stations in Beijing. (b) Attention weights of S_{21}.

Fig. 8. (a) Locations of stations, where only the discussed sensors are shown. S_{21} is the target sensor to compute attention weights. Note that stations S_{13}, S_{31} are extremely far from S_{21}. (b) Attention scores of station S_{21}.

VI. Conclusions and Future Works

In this paper, we propose a novel DualSTN model for spatiotemporal inference. To better learn the short- and long-term patterns, we decouple the model into two components: JST-GAT and SG-GRU. The first aims to capture delicate short-term joint spatiotemporal correlations while the second network focuses on long-term patterns by a time skip strategy. Extensive experiments on four real-world applications suggest that our DualSTN offers state-of-the-art performances against previous baselines. Further evaluations also justify the effectiveness of the modules as well as the interpretation ability brought by attention mechanisms.

In the future, we can explore ways to speed up the inference without losing performance, as we find that SG-GRU consumes a long inference time due to its recurrent structure. Then, the idea of joint attention can be transferred to the forecasting task for better capturing spatiotemporal relations. In addition, more complex models can be proposed to integrate forecasting and inference as a unified task.

Acknowledgments

This research is supported by Singapore Ministry of Education Academic Research Fund Tier 2 under MOE's official grant number T2EP20221-0023. It is also supported by Guangzhou Municipal Science and Technology Project 2023A03J0011.

References

[1] Y. Zheng, F. Liu, and H.-P. Hsieh, "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*, 2013, pp. 1436–1444.

[2] G. Y. Lu and D. W. Wong, "An adaptive inverse-distance weighting spatial interpolation technique," *Computers & geosciences*, vol. 34, no. 9, pp. 1044–1055, 2008.

[3] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer school on machine learning*. Springer, 2003.

[4] G. Appleby, L. Liu, and L.-P. Liu, "Kriging convolutional networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 3187–3194.

[5] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *2nd International Conference on Learning Representations, ICLR, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[6] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 1025–1035.

[7] Y. Wu, D. Zhuang, A. Labbe, and L. Sun, "Inductive graph neural networks for spatiotemporal kriging," in *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 2021, pp. 4478–4485.

[8] J. Han, H. Liu, H. Zhu, H. Xiong, and D. Dou, "Joint air quality and weather prediction based on multi-adversarial spatiotemporal networks," in *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 2021.

[9] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 753–763.

[10] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-gcn: A temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848–3858, 2019.

[11] S. Yang, M. Dong, Y. Wang, and C. Xu, "Adversarial recurrent time series imputation," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[12] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 914–921.

[13] C. Shang, J. Chen, and J. Bi, "Discrete graph structure learning for forecasting multiple time series," in *9th International Conference on Learning Representations*, 2021.

[14] Y. Qin, Y. Fang, H. Luo, F. Zhao, and C. Wang, "Dmgcrn: Dynamic multi-graph convolution recurrent network for traffic forecasting," *arXiv preprint arXiv:2112.02264*, 2021.

[15] M. Li and Z. Zhu, "Spatial-temporal fusion graph neural networks for traffic flow forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 5, 2021, pp. 4189–4196.

[16] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long-and short-term temporal patterns with deep neural networks," in *The 41st international ACM SIGIR conference on research & development in information retrieval*, 2018, pp. 95–104.

[17] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI*, 2019, pp. 1907–1913.

[18] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. K. Prasanna, "Graphsaint: Graph sampling based inductive learning method," in *8th International Conference on Learning Representations, ICLR*, 2020.

[19] N. Cressie, *Statistics for spatial data*. John Wiley & Sons, 2015.

[20] H. Saito and P. Goovaerts, "Geostatistical interpolation of positively skewed and censored data in a dioxin-contaminated site," *Environmental Science & Technology*, vol. 34, no. 19, pp. 4228–4235, 2000.

[21] J. Wallin and D. Bolin, "Geostatistical modelling using non-gaussian matérn fields," *Scandinavian Journal of Statistics*, 2015.

[22] X. Yi, Y. Zheng, J. Zhang, and T. Li, "ST-MVL: filling missing values in geo-sensory time series data," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI*, 2016, pp. 2704–2710.

[23] H. Ozkan, O. S. Pelvan, and S. S. Kozat, "Data imputation through the identification of local anomalies," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 10, pp. 2381–2395, 2015.

[24] M. T. Bahadori, Q. R. Yu, and Y. Liu, "Fast multivariate spatio-temporal analysis via low rank tensor learning," *Advances in neural information processing systems*, vol. 27, 2014.

[25] H.-F. Yu, N. Rao, and I. S. Dhillon, "Temporal regularized matrix factorization for high-dimensional time series prediction," *Advances in neural information processing systems*, vol. 29, 2016.

[26] M. Śmieja, Ł. Struski, J. Tabor, B. Zieliński, and P. Spurek, "Processing of missing data by neural networks," *Advances in neural information processing systems*, vol. 31, 2018.

[27] Y. Wu, D. Zhuang, M. Lei, A. Labbe, and L. Sun, "Spatial aggregation and temporal convolution networks for real-time kriging," *arXiv preprint arXiv:2109.12144*, 2021.

[28] W. Cheng, Y. Shen, Y. Zhu, and L. Huang, "A neural attention model for urban air quality inference: Learning the weights of monitoring stations," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[29] Q. Han, D. Lu, and R. Chen, "Fine-grained air quality inference via multi-channel attention model," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021*, 2021.

[30] 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.

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

[32] D. Xu, W. Cheng, D. Luo, X. Liu, and X. Zhang, "Spatio-temporal attentive rnn for node classification in temporal attributed graphs." in *IJCAI*, 2019, pp. 3947–3953.

[33] X. Shu, L. Zhang, Y. Sun, and J. Tang, "Host–parasite: Graph lstm-in-lstm for group activity recognition," *IEEE Transactions on neural networks and learning systems*, vol. 32, no. 2, pp. 663–674, 2020.

[34] F. Liu, J. Tian, L. Miranda-Moreno, and L. Sun, "Adversarial danger identification on temporally dynamic graphs," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[35] Y. Liang, K. Ouyang, J. Sun, Y. Wang, J. Zhang, Y. Zheng, D. Rosenblum, and R. Zimmermann, "Fine-grained urban flow prediction," in *Proceedings of the Web Conference 2021*, 2021, pp. 1833–1845.

[36] M. Li, S. Chen, Y. Shen, G. Liu, I. W. Tsang, and Y. Zhang, "Online multi-agent forecasting with interpretable collaborative graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[37] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *6th International Conference on Learning Representations, ICLR*, 2018.

[38] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187–197, 2015.

[39] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[40] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *6th International Conference on Learning Representations, ICLR 2018*, 2018.

[41] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 922–929.

[42] C. Zheng, X. Fan, C. Wang, and J. Qi, "Gman: A graph multi-attention network for traffic prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[43] G. Wang, R. Ying, J. Huang, and J. Leskovec, "Multi-hop attention graph neural networks," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021.

[44] L. Cai, K. Janowicz, G. Mai, B. Yan, and R. Zhu, "Traffic transformer: Capturing the continuity and periodicity of time series for traffic forecasting," *Transactions in GIS*, vol. 24, no. 3, pp. 736–755, 2020.

[45] R. Huang, C. Huang, Y. Liu, G. Dai, and W. Kong, "Lsgcn: Long short-term traffic prediction with graph convolutional networks." in *IJCAI*, vol. 7, 2020, pp. 2355–2361.

[46] F. Li, J. Feng, H. Yan, G. Jin, D. Jin, and Y. Li, "Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution," *arXiv preprint arXiv:2104.14917*, 2021.

[47] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems, NeurIPS*, 2020.

[48] Y. Shin and Y. Yoon, "Pgcn: Progressive graph convolutional networks for spatial-temporal traffic forecasting," *arXiv preprint arXiv:2202.08982*, 2022.

[49] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[50] Y. Liang, S. Ke, J. Zhang, X. Yi, and Y. Zheng, "Geoman: Multi-level attention networks for geo-sensory time series prediction," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI*, 2018, pp. 3428–3434.

[51] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[52] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *3rd International Conference on Learning Representations, ICLR*, 2015.

[53] M. Sengupta, Y. Xie, A. Lopez, A. Habte, G. Maclaurin, and J. Shelby, "The national solar radiation data base (nsrdb)," *Renewable and Sustainable Energy Reviews*, vol. 89, pp. 51–60, 2018.

[54] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE signal processing magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[55] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR*, Y. Bengio and Y. LeCun, Eds., 2015.

[57] P. Huang, J. Han, N. Liu, J. Ren, and D. Zhang, "Scribble-supervised video object segmentation," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 2, pp. 339–353, 2021.

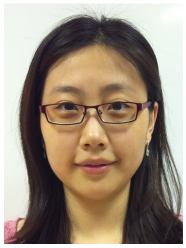
[58] D. Zhang, J. Han, L. Yang, and D. Xu, "Spftn: A joint learning framework for localizing and segmenting objects in weakly labeled videos," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 42, no. 2, pp. 475–489, 2018.

Junfeng Hu received the B.E. degree from the School of Big Data & Software Engineering, Chongqing, China, in 2020 and received the Master degree from the National University of Singapore, in 2022. He is currently a Ph.D. student in School of Computing at National University of Singapore. His research interests include spatio-temporal data mining and graph neural networks. He has published papers at conferences such as KDD, ECAI, ECML, IJCNN, and ICME.

Yuxuan Liang is an Assistant Professor at Intelligent Transportation Thrust, Hong Kong University of Science and Technology (Guangzhou). He is currently working on the research, development, and innovation of spatio-temporal data mining and AI, with a broad range of applications in smart cities. Prior to that, he obtained his PhD degree at NUS. He published over 40 peer-reviewed papers in refereed journals and conferences, such as KDD, WWW, NeurIPS, ICLR, ECCV, AAAI, IJCAI, Ubicomp, and TKDE. Those papers have been cited over 2,000 times (Google Scholar H-Index: 21). He was recognized as 1 out of 10 most innovative and impactful PhD students focusing on data science in Singapore by Singapore Data Science Consortium (SDSC).

Zhencheng Fan is a Ph.D. candidate at the University of Technology Sydney, Australia, specializing in memristor-based neural networks, data mining, and computer vision. He received his Bachelor's degree in Software Engineering from Chongqing University, China in 2020. His work has been published in conferences including KDD, ECAI, and KSEM.

Li Liu is a professor at Chongqing University. He is also serving as a Senior Research Fellow of School of Computing at the National University of Singapore. Li received his Ph.D. in Computer Science from the Université Paris-sud XI in 2008. His research interests are in pattern recognition, data analysis, and their applications on human behaviors. He aims to contribute in interdisciplinary research of computer science and human related disciplines. Li has published widely in conferences and journals with more than 100 peer-reviewed publications. Li has been the Principal Investigator of several funded projects from government and industry.



Yifang Yin received the B.E. degree from the Department of Computer Science and Technology, Northeastern University, Shenyang, China, in 2011, and received the Ph.D. degree from the National University of Singapore, Singapore, in 2016. She is currently a scientist at Institute for Infocomm Research, A*STAR, Singapore. She also holds an adjunct faculty position at IIIT-Delhi. Before joining A*STAR, she worked as a senior research fellow with the Grab-NUS AI Lab at the National University of Singapore. She also worked as a Research Intern at the Incubation Center, Research and Technology Group, Fuji Xerox Co., Ltd., Japan, from October, 2014 to March, 2015. Her research interests include machine learning, spatiotemporal data mining, and multimodal analysis in multimedia.

Roger Zimmermann received his M.S. and Ph.D. degrees from the University of Southern California (USC) in 1994 and 1998, respectively. He is currently a Professor with the Department of Computer Science at the National University of Singapore (NUS) and a key investigator with the Grab-NUS AI Lab. He previously was a Deputy Director with the Smart Systems Institute (SSI) and co-directed the Centre of Social Media Innovations for Communities (CoSMIC) at NUS. He has co-authored a book, seven patents, and more than 300 conference publications, journal articles, and book chapters. His research interests include streaming media architectures, distributed systems, mobile and geo-referenced video management, applications of machine/deep learning, and spatial data management. He is an associate editor for IEEE MultiMedia, ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), Springer Multimedia Tools and Applications (MTAP), and IEEE Open Journal of the Communications Society (OJ-COMS). He is a distinguished member of the ACM and a senior member of the IEEE. Further information can be found at http://www.comp.nus.edu.sg/~rogerz/.