Spatial-Temporal Aware Inductive Graph Neural Network for C-ITS Data Recovery

Wei Liang, Yuhui Li, Kun Xie, *Member, IEEE*, Dafang Zhang, Kuan-Ching Li, *Senior Member, IEEE*, Alireza Souri, *Senior Member, IEEE*, and Keqin Li, *Fellow, IEEE*

***Abstract*— With the prevalence of Intelligent Transportation Systems (ITS), massive sensors are deployed on roadside, vehicles, and infrastructures. One key challenge is imputing several different types of missing entries in spatial-temporal traffic data to meet the high-quality demand of data science applied in Cooperative-ITS (C-ITS) since accurate data recovery is critical to many downstream tasks in ITSs, such as traffic monitoring and decision making. For such, it is proposed in this article solutions to three kinds of data recovery tasks in a unified model via spatial-temporal aware Graph Neural Networks (GNNs), named Spatial-Temporal Aware Data Recovery Network (STAR), enabling a real-time and inductive inference. A residual gated temporal convolution network is designed to permit the proposed model to learn the temporal pattern from long sequences with masks and an adaptive memory-based attention model for utilizing implicit spatial correlation. To further exploit the generalization power of GNNs, a sampling-based method is adopted to train the proposed model to be robust and inductive for online servicing. Extensive numerical experiments on two real-world spatial-temporal traffic datasets are performed, and results show that the proposed STAR model consistently outperforms other baselines at 1.5-2.5 times on all kinds of imputation tasks. Moreover, STAR can support recovery data for 2 to 5 hours, with its performance barely unchanged, and has comparable performance in transfer learning and time-series forecast. Experimental results demonstrate that STAR provides adequate performance and rich features for multiple data recovery tasks under the C-ITS scenario.**

***Index Terms*— Cooperative intelligent transportation system, data recovery, graph neural network, spatial-temporal.**

I. Introduction

With the advancement of communication and information security technologies [1], smart cities are rapidly growing the scope and coverage of sensor networks to collect and analyze data for city management such as traffic systems, urban security, and weather forecast. With the widespread of sensors of all types, a massive volume of data is generated [2] and thereby, leading to possible advanced data science technologies applied in smart city applications. One of the most successful applications is Intelligent Transportation Systems (ITS), which broadly supports mitigating traffic congestion, improving road safety, increasing road capacity, and saving fuel consumption using data analysis algorithms. As illustrated in Figure 1, Cooperative-ITS (C-ITS) has emerged to enable multiple isolated ITS to cooperate with each other in recent years, thereby further improving safety, sustainability, efficiency, and comfort by exploiting advanced communication and collaboration between standalone agents.

As the volume of C-ITS systems and wireless communication networks expands, cases of sensor malfunction, transmission interruption, and missing data have become inevitable issues, and therefore, severe consequences may occur. For instance, such a phenomenon may lead to erroneous conclusions, as missing values may distort statistical characteristics and cause a model to produce unexpected results, misleading wrong decisions. In addition, deploying sensors in urban areas is expensive and laborious, not to mention the increasing system operation and maintenance costs. As a matter of fact, only a limited number of sensors is available for the C-ITS to retrieve a conspectus of the region. Hence, the data recovery[1] task is critical, since many applications may rely on it.

Essentially, the missing patterns can be summarized into three types, namely random missing, segment missing, and blockout missing, and corresponding intuitive examples of data missing patterns are presented in Figure 2. Random missing may cause accidental packet loss; segment missing may indicate malfunctioning, and blockout missing is due to the new deployment of sensors. In practice, all three kinds of data

Manuscript received 2 August 2021; revised 9 November 2021 and 7 February 2022; accepted 22 February 2022. Date of publication 14 March 2022; date of current version 2 August 2023. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFA1000600, in part by the National Natural Science Foundation of China under Grant 62072170 and Grant 61976087, in part by the Science and Technology Project of Department of Communications of Hunan Provincial under Grant 202101, in part by the Key Research and Development Program of Hunan Province under Grant 2022GK2015, and in part by the Hunan Provincial Natural Science Foundation of China under Grant 2021JJ30141. The Associate Editor for this article was W. Wei. *(Corresponding author: Kuan-Ching Li.)*

Wei Liang is with the School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China, also with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China, and also with the Hunan Key Laboratory for Service Computing and Novel Software Technology, Xiangtan, Hunan 411201, China.

Yuhui Li, Kun Xie, and Dafang Zhang are with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China.

Kuan-Ching Li is with the School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China (e-mail: kuancli@outlook.com).

Alireza Souri is with the Department of Computer Engineering, Haliç University, 34394 Istanbul, Turkey.

Keqin Li is with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA.

Digital Object Identifier 10.1109/TITS.2022.3156266

[1] Data recovery and data imputation are used interchangeably in this article.

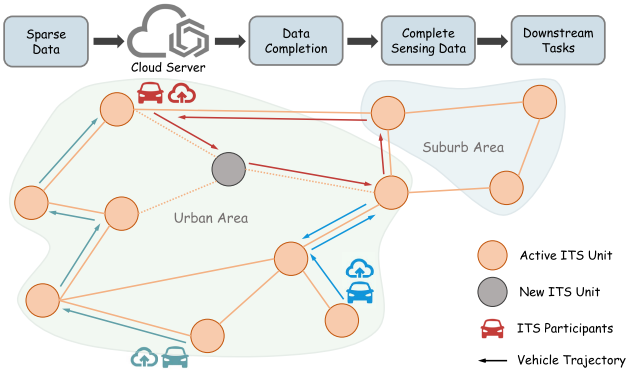


Fig. 1. The demonstration of data recovery workflow in cooperative intelligent transportation system.

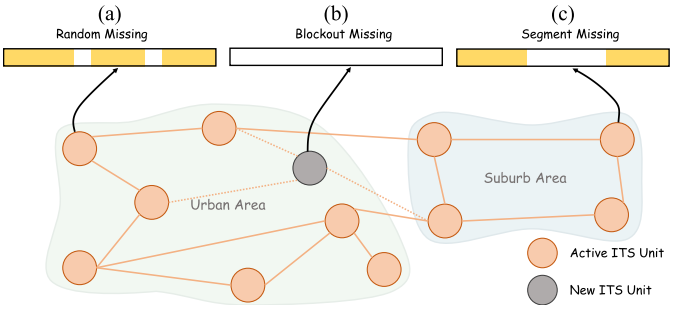


Fig. 2. Data missing patterns of traffic data. (a) **Random Missing** is caused by unexpected transmission errors, and interpolation methods can quickly fill the missing values. (b) **Segment Missing** is caused by power outages, sensor malfunctioning, and extreme weather conditions. Factorization-based methods and neural network-based models can fill these missing values. (c) **Blockout Missing** is caused by new deployments or long-time failure, as filling missing values for such situations may be challenging, given that no historical data is available, so thus, nearby sensors are used to fill the need to handle the complicated spatial-temporal dependencies.

missing patterns co-exist in real-world collected sensor data, incurring additional difficulties to data science. If missing data is accurately reconstructed, this is an undoubtedly valuable support for autonomous driving, traffic flow prediction, and deploying virtual sensors.

Unfortunately, this is not an easy task, and there are needs and challenges to design a highly precise while fast algorithm:

- **Fast and Accurate Data Recovery**. The algorithm should fill the missing values as soon as possible to meet the real-time requirement of several subsequent tasks. The model should be inductive to get the imputed data, which means no retraining when new data arrives. Matrix/Tensor completion methods are mostly transductive, which means they cannot generalize to unseen nodes (spatial aspect). In addition, completion-based methods are also unable to generalize to the next time-window (temporal aspect).
- **Irregular Missing Pattern**. Due to the randomness of failure cases and data packet loss, the missing patterns are usually highly irregular, and the total sampling rate varies. It causes difficulty in representation learning for such data dynamic scenes.

The fundamental challenge of the data completion task is to exploit the limited observed data, using the internal spatial-temporal correlations, to impute the missing entries effectively. Although significant progress has been made on spatial-temporal aware time series forecast in recent years, a few numbers of literature focuses on the neural network-based spatial-temporal imputation problem with complex missing patterns. In this article, inspired by the successful application of [3], [4] that GNNs are promising tools for inductive tasks, we address the challenges mentioned above and propose a novel framework named **S**patial-**T**emporal **A**ware Data **R**ecovery Network (STAR) for this task based on Graph Neural Networks (GNNs). The technical contributions are threefold:

- We propose a novel inductive spatial-temporal model called STAR to solve the data imputation problem under C-ITS. Compared with transductive methods, the proposed model can meet the requirements of real-time traffic data imputation without retraining the whole model,
- The proposed model can capture spatial-temporal dependencies with semantics effectively and efficiently. The core idea is to assemble an adaptive memory-based attention network into graph convolution and utilize dilated Temporal Convolution Network (TCN) to accelerate training and inference,
- To conduct extensive numerical experiments on real-world sensor datasets to verify the performance of the proposed model.

The remainder of this article is organized as follows. Section II briefly reviews related works, Section III presents the methodology, Section IV discusses the experiment results of the proposed model, and finally, concluding remarks and future directions are presented in Section V.

II. Related Work

A. C-ITS

ITS integrates multiple highly trended advanced technologies, including sensors network, communication, control theory, and artificial intelligence. It focuses on digital technologies that provide intelligence for systems. The prevalence of these systems and emerging network technologies (e.g., 5G, WiFi6, Internet of Things (IoT), SD-WAN) enable C-ITS. Infrastructures equipped with C-ITS can cooperate to improve overall system efficiency, reliability, and sustainability. For example, Ref. [5] proposed an augmented vehicle localization that combined global navigation satellite systems (GNSS) with vehicle-to-anything (V2X) communication systems. Reference [6] exploited streaming C-ITS data to detect anomaly stopped cars and a growing pothole on the road using concept drift detection methods. Reference [7] proposed a deep neuro-evolution model to implement a cooperative control scheme that integrated ramp metering, speed limits, and lane change control agents to improve freeway traffic. Reference [8] introduced a choreography-based heterogeneous service composition platform to accelerate the reuse-based development of an urban traffic coordination application.

Despite the outstanding achievement, some open issues that hinder the application of data science for C-ITS still exist [9]. This article focuses on the data imputation problem. That is,

every single component collects traffic data and uses wireless communication to propagate messages. With the increasing volume of communication systems, data transmission errors and data missing become assignable. In addition, as a critical component of the system, sensors still require high costs to deploy to large-scale networks [10]. Fortunately, these two problems can be alleviated by a well-designed spatial-temporal aware data recovery algorithm, and so thus, a better model for high accuracy data recovery and estimation under C-ITS is urgently needed.

B. Traffic Flow Forecast

The traffic flow forecasting problem is a fundamental yet challenging issue. Earlier works as those presented in [11]–[13] attempted to treat it as a time-series prediction problem in isolated points. Unfortunately, these methods heavily depend on local seasonality features, and hence they often fail to model interstation dependencies. Recent works explore the power of GNNs in modeling spatio-temporal data. References [14], [15] proposed RNN-based methods that capture spatial and temporal dependency using graph convolution and recurrent neural networks, respectively. Other alternatives, as presented in [16], [17], are equipped with stacked CNN-based temporal encoder and graph convolution-based spatial encoder to gain better representation and faster training speed. Li *et al.* [18] summarize and benchmark the previous works on traffic flow forecast, then proposing novel RNNs with dynamic graph inputs on each step.

C. Spatial-Temporal Kriging for Blockout Missing

Gaussian process regression (GPR) [19], [20] is an effective tool to solve the Kriging problem, as it applies a flexible kernel to construct spatiotemporal correlations. Nevertheless, the major drawback of GPR is the high computation overhead, which limits its real-time application.

In recent years, neural network-based Kriging emerged. Reference [4] overcame strong Gaussian assumptions and directly used neighboring observations when generating predictions. Ref. [21] proposed a novel generative adversarial network for recovering missing entries in a fixed-size matrix, and finally, Reference [3] applied diffusion graph convolution and exploited training technique to enable inductive inference. Unfortunately, most of the models mentioned above are transductive. That is, they needed to retrain the entire model when the network structure is changed even slightly. Some recent studies [3], [22]–[24] demonstrated that GNN could generalize to an unseen new structure of graphs (i.e., new nodes or new edges introduced). Inspired by these works, we develop an inductive model to solve the spatial-temporal Kriging problem for dynamic C-ITS.

D. Spatial-Temporal Imputation for Non-Blockout Missing

Works in literature pointed out the spatial-temporal imputation problem as matrix/tensor completion, as they leverage the road network structure as regularization under the matrix completion framework [25]–[27]. To further utilize more spatial-temporal patterns, other approaches as [28]–[30] tried tensor factorization to reconstruct the traffic data tensor, implicitly learning latent factors for representing spatial and temporal correlation. For example, [31] proposed a Bayesian Tensor Factorization model, [32] leveraged autoregressive in tensor completion to capture strong temporal correlation in traffic data, and [33] optimize nuclear norm minimization through integrating linear unitary transformation, achieving high scalability. However, low-rank matrix/tensor completion methods have two significant drawbacks. The former is, a retrain is required if it is needed to impute a new sparse tensor, inducing severe time-complexity concerns. On the other hand, low-rank constraints and linearity may force the model to capture a smooth pattern, limiting it to capture highly complex internal temporal and spatial patterns.

TABLE I
MATHEMATICAL SYMBOLS AND DESCRIPTION

Notation	Description
$\mathbf{X}$	traffic data matrix
$\hat{\mathbf{X}}$	imputed data matrix
$\mathbf{M}$	mask, indicating whether the entry is zero
$\mathbf{G}$	graph generated from the distance matrix
$f(\cdot)$	model function
$\mathbf{W}$,b	trainable parameters of a layer
$\mathbf{\Theta}$	trainable parameters of the entire model
$\mathbf{A}_f, \mathbf{A}_b$	forward and backward adjacent matrix calculated via $\mathbf{G}$
$\mathbf{H}^{(l)}$	the output of l-th layer

III. METHODOLOGY

In this section, the definition of the spatio-temporal imputation problem in math is formally presented. First, three building blocks: temporal, spatial, and diffusion graph convolution blocks are designed, and next, we outline the inductive architecture of the proposed model to show how sub-modules iterate together to solve the data recovery problem.

A. Notation

The mathematical symbols used in this section are presented in the following table.

B. Problem Description

Spatial-temporal imputation problem under C-ITS scenario refers to interpolating missing data for target sensor according to sampled sensor data. Initially, we denote the entire sensor network with N nodes and E edges as graph G while the sampled data $\mathbf{X} \in \mathbb{R}^{N \times T}$, where a mask $\mathbf{M}$ is created to indicate the non-zero entries in $\mathbf{X}$. Next, after n new nodes with e new edges related to them are added to the sensor network $\mathbf{G}$, we have a new graph $\mathbf{G}'$. Notably, n new nodes only have e edges as knowledge prior. Thus, our task is to interpolate $\mathbf{X}' \in \mathbb{R}^{(N+n) \times T}$ according to both $\mathbf{G}'$ and $\mathbf{X}$ by estimating the missing history data for n nodes. Therefore, we formulate the data imputation task as function f:

$$\begin{aligned} &\mathbf{X}' = f(\mathbf{X}, \mathbf{M}, \mathbf{G}') \\ &s.t.\ \mathbf{X} * \mathbf{M} = \mathbf{X}' * \mathbf{M} \end{aligned} \tag{1}$$

According to the above formulation, we treat the data imputation task as a conditional generation problem using mask $\mathbf{M}$.

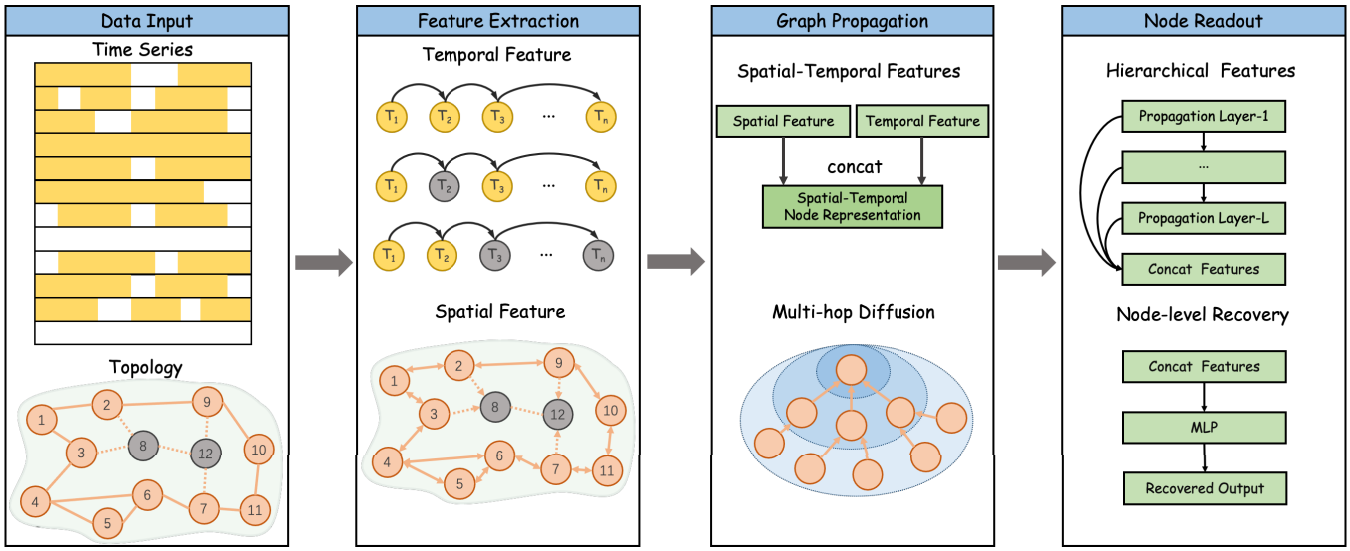


Fig. 3. The framework of STAR.

C. Framework of STAR

We present the framework of STAR in Figure 3. It consists of two parallel feature extraction modules, graph convolution layers, and output layers. By stacking multiple graph convolution and TCN layers, our model can handle spatial-temporal dependencies at a different scale. For example, we can stack more TCN layers if the input time series is long and contains more graph convolution layers to capture long-range spatial dependencies.

D. Temporal Feature Extraction

Recurrent Neural Network (RNN)-based approaches are applied to extract features of a sequence such as time series and natural language. However, RNN-based approaches do have disadvantages, and they are threefold. First, they cannot handle long sequences, since memory may lose. Second, they suffer from gradient vanish/explosion problems, and finally, the latter one, the recursive computation manner, brings low efficiency in parallel training and inference. With the concern mentioned above, we adopt TCN [34] in our tasks instead of RNN. As illustrated in Figure 4, TCN applies dilated causal convolution, which can enlarge its receptive fields exponentially and thus enable the proposed model to capture long-range temporal patterns, as well as to save computation resources.

Inspired by gating mechanisms in RNNs and GLU [35], we use residual gated TCN (RG-TCN) to control information flow more effectively in a deep network. First, we stack corrupted data series and masks to form original input $\mathbf{H}^{(0)}$:

$$\mathbf{H}^{(0)} = stack(\mathbf{X}, \mathbf{M}, \mathbf{1} - \mathbf{M}). \tag{2}$$

Given the $\mathbf{H}$ as input, RG-TCN takes the form:

$$\begin{aligned} \mathbf{H}' &= tanh(\mathbf{W}_1 \star \mathbf{H}^{(l)} + b_1) \odot sigmoid(\mathbf{W}_2 \star \mathbf{H}^{(l)} + b_2) \\ \mathbf{H}^{(l+1)} &= \mathbf{H}' + \phi(\mathbf{H}^{(l)}), \end{aligned} \tag{3}$$

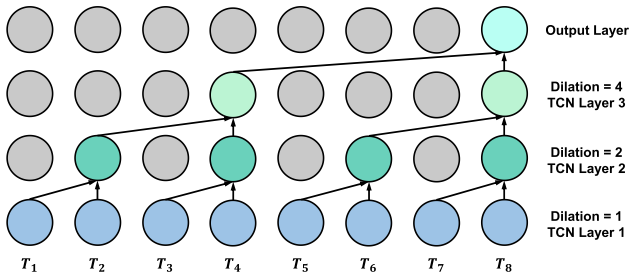


Fig. 4. TCN with kernel=2, stride=2, dilation=2.

where $\mathbf{W}_1, \mathbf{W}_2, b_1, b_2$ are learnable parameters, $tanh(\cdot)$ and $sigmoid(\cdot)$ are two commonly used activation functions; and ϕ denotes 1D-Conv with 1×1 kernel.

The missing values positions are crucial for imputation tasks. We notice that, if we input a corrupted time series into the neural network after the min-max scaler, the missing values are set to zero, making it difficult to distinguish small values and missing values. The mask, indicating the missing values, contains positional information that guides the model to extract temporal patterns from other time slices. The architecture of the temporal feature extraction module is presented in Figure 5.

The proposed RG-TCN will not change the input length of the time series data but changes the channel depth during the hidden layers. Therefore, we maintain the identical length of data after being processed by the RG-TCN.

E. Attention-Based Spatial Feature

To extract spatial features for further fusion, we propose an attention-based spatial module that combines TCN, graph convolution, and attention mechanism with linear time complexity and space complexity. The architecture of this module is shown in Figure 6.

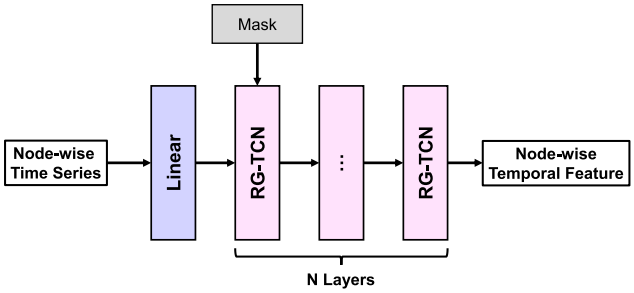


Fig. 5. The architecture of temporal feature extraction module.

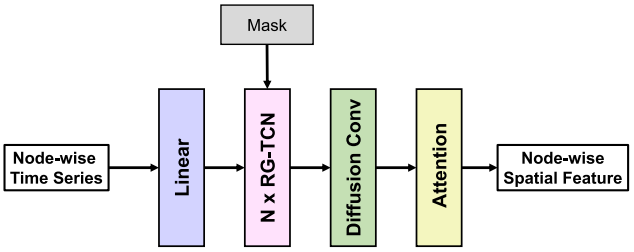


Fig. 6. Architecture of spatial feature extraction module.

In this module, we first feed node-wise time series into RG-TCN to extract its temporal patterns and then use graph convolution to obtain node-wise embedding. The graph convolution aims to obtain embedding for new nodes while aggregating neighborhood information to distinguish them from one another. Once node-level representation is computed, we use the attention module to capture global similarity even two nodes are in different connected components.

Recent works on spatial-temporal traffic prediction, as those presented in [14], [16], [17], apply graph convolution to model spatial correlations. However, due to the over-smooth problem, we cannot stack graph convolution layers many times to capture long-range dependencies, as nodes can only capture signals from a local sub-graph. Besides, no path is even available to connect sensors with a similar pattern. Therefore, we argue that graph convolution is insufficient to capture spatial correlation thoroughly. It is proposed in [17] a self-adaptive adjacency matrix to solve such a problem. However, this solution's major drawback is that it fails to generalize unseen nodes, which belongs to transductive methods. One straight-forward inductive solution is self-attention, while it suffers from $O(n^2)$ computation complexity and only captures correlation inside given nodes. Considering the rapidly expanding network scale and complicated spatial-temporal dependencies, we adopt external attention [36]. The linear complexity and global sample-wise memory can significantly facilitate the real-time data imputation. External attention module takes the form:

$$\mathbf{H}^{(l+1)} = EA(\mathbf{H}^{(l)}) = Norm(\mathbf{H}^{(l)}\mathbf{M}_k^T)\mathbf{M}_v, \tag{4}$$

where $\mathbf{M}_k^T$ and $\mathbf{M}_v$ are two learnable parameter matrices as memories for key-value matching, $Norm(\cdot)$ is a two-stage normalization function that computes Softmax and L1-norm in sequence.

F. EA-Diffusion Convolution and Output Layer

The temporal and attention-based spatial feature modules are two branches for spatial-temporal feature extraction. We concatenate these two representations as node-level embedding for further propagation inside the graph.

The real-world sensor networks have underlying directed topology. For example, sensors are deployed on the road, which naturally forms a bi-directed graph. We adopt diffusion graph convolution networks (DGCN) [14] as the propagation layer to handle this directed graph. DGCN treats forward edges and backward edges separately to create two matrices–forward transition matrix $\mathbf{A}_f$ and backward transition matrix $\mathbf{A}_b$. We denote the diffusion steps as K, the diffusion graph convolution layer is written as:

$$\mathbf{H}^{(l+1)} = \sum_{k=0}^{K}(\mathbf{A}_f^k\mathbf{H}^{(l)}\mathbf{W}_{k1} + \mathbf{A}_b^k\mathbf{H}^{(l)}\mathbf{W}_{k2}), \tag{5}$$

where transition matrix $\mathbf{A}_f$ and $\mathbf{A}_b$ are generate through $\mathbf{A}_f = \mathbf{A}/\sum_j \mathbf{A}_{ij}$, $\mathbf{A}_b = \mathbf{A}^T/\sum_j \mathbf{A}_{ij}^T$.

Graph Neural Networks highly rely on the pre-defined adjacent matrix, given that it limits the neural network to capture semantic similarity inside a large-scale sensor network. In addition, the demand for semantic similarity depends on the dataset itself rather than the network structure. Besides, other works use attention mechanisms [37] and trainable adaptive adjacent matrices [17], [38], [39] to capture semantic similarity. However, the former suffers from high computation overhead, while the latter cannot generalize to unseen nodes. To tackle the abovementioned challenges, we designed an external attention-enhanced diffusion convolution to learn semantic similarity adaptively:

$$\mathbf{H}^{(l+1)} = \alpha * EA(\mathbf{H}^{(l)}) + \sum_{k=0}^{K}(\mathbf{A}_f^k\mathbf{H}^{(l)}\mathbf{W}_{k1} + \mathbf{A}_b^k\mathbf{H}^{(l)}\mathbf{W}_{k2}), \tag{6}$$

where α is initially set to zero as a weight to control semantic similarity learning, and $EA(\cdot)$ is introduced in Equation 4. Through this design, we enhanced diffusion convolution with a second branch of linear time complexity semantic similarity learning.

To better utilize features at multiple-scale and accelerate the training process, we adopt a concatenation for node features produced by each layer. In this layout, the neural network can extract specific N-hop neighborhood information for data recovery. Besides, the residual connection is added to enable the information and gradient to flow through the whole network. The graph convolution, as well as the output layers, are presented in Figure 7.

G. Training and Loss Function

As mentioned in subsection III-B, our task is to reconstruct the missing sensor data. Intuitively, we can define the loss functions focusing only on the reconstructing errors used in

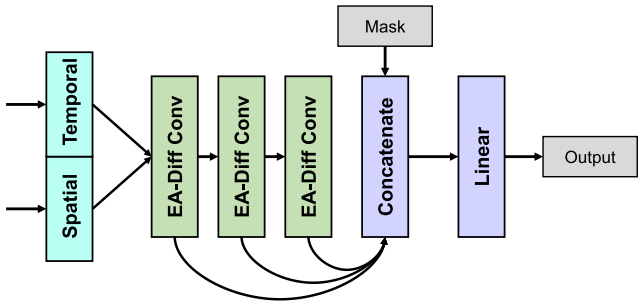


Fig. 7. The architecture of graph convolution layers and output layers.

unsupervised tasks (e.g., mask language model, autoencoder). Compared with a loss function with only the masked data, the reconstruction error enables the model to generalize to unseen samples better. The reconstruct error is given as below:

$$\mathcal{L} = \|\mathbf{X} - \hat{\mathbf{X}}\|_2^2 + \lambda||\Theta||_2, \tag{7}$$

where Θ denotes all trainable parameters, and λ is the regularization coefficient empirically set to 0.01.

To learn generalized graph convolution and adapt to new network structures, we use sampling-based training strategies, such as [3], [22], [24]. As given in Algorithm 1, we randomly treat a part of nodes as observed and the rest as Blockout missing nodes for imputation for each batch in this training algorithm. In addition, to further improve the robustness and enable support of multiple recovery tasks, we generate two other kinds of missing masks to simulate real-world data corrupted scenes.

Algorithm 1: Training Pseudocode (PyTorch-Style)

Input: Iteration *num_iter*,
number of batches *num_batch*,
batch size *batch_size*,
timespan *span*,
number of masked nodes *num_masked*,
training dataset $\mathbf{X} \in \mathcal{R}^{n \times T}$,
adjacent Matrix for training **A**,
model to be trained *model*
Output: trained model *model*

```
for i=1:num_iter do
    for n=1:num_batch do
        /* Prepare Training Data */
        batch, label_data = list(), list()
        spts = randInt([0, T-span], batch_size)
        observed = randSample([0, n], n-num_masked)
        for bs=1:batch_size do
            batch.append(X[observed,spts[bs]: spts[bs] +
             span])
            label.append(X[:,spts[bs]: spts[bs] + span])
        /* Prepare Mask for Data Missing */
        randMask = genRandMask(batch.shape)
        segMask = genSegMask(batch.shape)
        blockMask = genBlockMask(batch.shape)
        mask = randMask * segMask * blockMask
        /* Model Inference and Optimization */
        A_f = forwardTransitionMatrix(A)
        A_b = backwardTransitionMatrix(A)
        optimizer.zero_grad()
        yhat = model(batch, mask, A_f, A_b)
        loss = criterion(yhat, label)
        loss.backward()
        optimizer.step()
return model;
```

IV. Experiments

In this section, we introduce the experiment environment, also evaluating the proposed model with an extensive number of experimentations.

A. Dataset

Two public spatial-temporal datasets are utilized to verify the proposed model. METR-LA records four months of traffic speed data on the highways of Los Angeles, California, USA through 207 sensors, and PEMS-Bay collects traffic speed data in California, USA, and normalized by a min-max scaler. We randomly select 75% of sensors for training and hold out the rest of 25% of sensors as testing data. Besides, we split the datasets in chronological order, and the portion of the train set and test set is 7:3. Detailed statistical properties of two datasets are presented in Table II.

TABLE II
Statistical Properties of Traffic Datasets

Dataset	Interval	#Timesteps	#Nodes	#Edges	Time Span
METR-LA	5min	34272	207	1515	4 months
PEMS-Bay	5min	52116	325	2691	6 months

B. Baselines

We compare our model with the following baselines:

- **Average**, takes the average values from its neighborhoods as the prediction.
- **2D-Krige**, which is provided by a Python framework downloaded from https://github.com/GeoStat-Framework/PyKrige for statistical simulations in geography. This method is only available when sensor locations are given.
- **GCN**, which introduces non-linearity compared with the average model. It aggregates neighborhood information under the message passing framework.
- **IGNNK** uses stacked diffusion graph convolution layers and applies training strategy to be inductive for spatial Kriging task.
- **STAR**. Whether the sub-modules are enabled or not, there have three variants–STAR-T only enables the temporal feature extraction module, STAR-S only enables the spatial feature extraction module, and STAR enables both modules for spatial-temporal feature extraction.

We also classify the baseline based on model category, spatial dependency modeling, temporal dependency modeling, and multistep imputation, as shown in Table III.

TABLE III
SUMMARY OF MODEL USED IN EXPERIMENT

Model	Category	Spatial	Temporal	Multistep
Average	Non Neural Network	✓	×	×
OrdinaryKrige	Non Neural Network	✓	×	×
GCN	Graph Neural Network	✓	×	✓
IGNNK	Graph Neural Network	✓	×	✓
STAR	Graph Neural Network	✓	✓	✓

C. Settings

We implement our model in PyTorch 1.7.1 with Python 3.7 and deploy it on a server equipped with Intel i9-9900KS process, 32GB memory, and an NVIDIA GTX 2080Ti GPU. For hyperparameters, we select 100 as the hidden dimension for linear mapping. To learn the long-range temporal pattern, we use six layers of RG-TCN with dilation factors 1,2,1,2,1,2 with kernel size 2 and stride 1. The activation and normalization layers are Leaky ReLU [40] and Layer Normalization [41], respectively. For the gradient descent algorithm, we select Adam optimizer [42]. The batch size is set to 8, and the learning rate is fixed to 0.008.

D. Metrics

To quantity our model performance and compare with other baseline methods, we choose the following three metrics:

- **MAE** (Mean Absolute Error). It is commonly used in evaluating the performance of regression tasks.

$$MAE = \frac{\sum |x_{ij} - \hat{x}_{ij}|}{N_{sample}}. \tag{8}$$

- **RMSE** (Root Mean Squared Error). RMSE is used to illustrate the degree of dispersion of the sample. For nonlinear fittings, smaller RMSE indicates better regression accuracy.

$$RMSE = \sqrt{\frac{\sum (x_{ij} - \hat{x}_{ij})^2}{N_{sample}}}. \tag{9}$$

- **MAPE** (Median Absolute Percentage Error). MAPE is used to estimate relative absolute error. It takes the form:

$$MAPE = \sum \left| \frac{x_{ij} - \hat{x}_{ij}}{x_{ij}} \right| \times 100\%. \tag{10}$$

E. Imputation Performance

In this section, we compare the proposed model with other baselines in different conditions of data missing to demonstrate the superiority of the proposed model. First, we set the random missing ratio to 20%, and then remove 200 segments of 30 minutes in each sensor for both the trainset and the rest. Next, we hold out 25% of sensors as unsampled. Finally, the experiment results are given in Table IV.

According to the results, we have the following conclusions:

1) High Performance: We compare the proposed model with two mathematical models and two GNNs. The proposed model consistently outperforms the baseline methods by a large margin in all kinds of imputation tasks. We identified that directly using neighborhood sensors can achieve competitive performance because of the strong correlation and impact between nearby sensors. The naive multi-layered GCN is also a firm baseline in imputation. Specifically, for random missing and segment missing tasks, we significantly outperform IGNNK, which shows that our scheme effectively captures the spatial-temporal context.

2) Robustness: We evaluate the performance to impute the highly corrupted data when three kinds of data missing coexist. Out model achieves MAE of 2.63, 4.74 in PEMS-Bay and METR-LA, respectively, which is a 44% and 29% improvement compared to the best baseline model. The high imputation performance under such highly corrupted inputs shows the strong robustness of our model. We also observed that when we applied the sampling training algorithm on IGNNK, it became precarious to blockout missing tasks because it does not apply any solution to the missing entries. Our model applies positive masks and negative masks to indicate valid and missing entries, treat missing value positions as useful information, and thus, achieve robustness and accuracy.

3) Flexibility: The proposed model is trained with randomly missing data and imputes them according to the given mask, and so it can support three types of missing data imputation in one single model. Moreover, according to the Table IV, we learned that the proposed model achieves highly competitive performance in all kinds of imputation tasks, which a feature can support the ITS to reduce the cost of the entire model life-cycle significantly.

F. Impact of Window Size

Table V presents the imputation accuracy of the STAR model, and other baseline approaches for 24-, 36-, 48-, 60-step (2 hours to 5 hours with the step of 1 hour) data recovery tasks on METR-LA and Seattle Highway datasets. The STAR model obtains the best recovery accuracy under nearly all evaluation metrics, except RMSE, for all horizons, thereby providing the effectiveness for spatial-temporal aware data recovery tasks.

From the experimental results, we conclude three significant features of the proposed model:

1) High Recovery Accuracy: The proposed model, which extracts the temporal features, performs better than other methods like IGNNK and Average. For example, for the 24-step recovery, STAR outperforms IGNNK by 20.7% and 11.7% on METR-LA and PEMS-Bay, respectively. The MAPE errors of the STAR are significantly lower than those of IGNNK. This phenomenon is mainly due to the ignorance of internal temporal patterns.

2) Spatio-Temporal Recovery Capability: To prove the STAR model can capture spatial and temporal dependencies, we compare the variants of the STAR model with IGNNK. As shown in Figure Figure 8(a), methods with temporal feature extraction have better recovery precision than baseline ones, indicating that our temporal module can capture temporal patterns from traffic data. Furthermore, according to Figure 8(b), we learn that by enabling spatial attention, RMSE errors decrease, suggesting that our proposed module captures long-range spatial correlation beyond pre-defined graph structure. Finally, only exploiting spatial and temporal features could reach the best performance, indicating the presence of spatial-temporal dependencies.

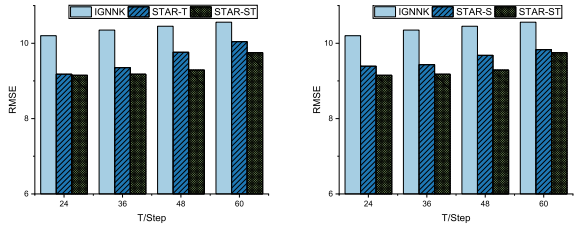
TABLE IV
MODEL PERFORMANCE UNDER DIFFERENT IMPUTATION TASKS

		Random Missing			Segment Missing			Blockout Missing			All		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
PEMSBay	Neighbour Mean	14.21	17.54	24.2%	6.41	8.67	13.3%	5.16	7.15	12.0%	5.73	7.91	13.4%
	2D-Krige	-	-	-	-	-	-	-	-	-	-	-	-
	GCN	4.01	5.95	7.9%	4.93	7.66	9.5%	5.61	10.92	15.2%	6.23	10.38	13.9%
	IGNNK	4.07	6.28	9.0%	5.22	7.80	11.4%	11.03	13.22	22.8%	4.74	7.18	10.3%
	STAR	1.63	2.57	3.3%	2.19	3.66	4.8%	3.52	6.09	8.3%	2.63	4.88	6.1%
METR	Neighbour Mean	14.35	17.58	28.0%	9.48	12.26	22.0%	6.59	9.20	17.1%	6.67	9.46	16.5%
	2D-Krige	7.87	10.21	22.8%	7.89	9.91	22.9%	8.77	11.21	24.4%	8.38	11.07	24.1%
	GCN	5.75	8.10	15.3%	7.85	10.35	18.3%	7.94	13.22	25.8%	8.42	12.83	24.1%
	IGNNK	6.19	8.75	16.2%	7.03	9.80	18.2%	15.38	18.29	38.1%	7.75	10.49	19.4%
	STAR	3.73	5.85	9.6%	4.53	6.96	12.1%	5.71	8.50	14.4%	4.74	7.41	12.3%

[1] We are unable to present 2D-Krige results on PEMS-Bay as sensor locations are not provided.

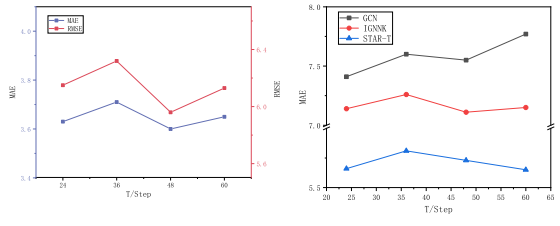
TABLE V
PERFORMANCE COMPARISON WITH DIFFERENT TIMESTEPS

	META-LA T=24			META-LA T=36			META-LA T=48			META-LA T=60		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
GCN	7.41	12.93	24.18%	7.60	12.50	23.83%	7.55	12.62	23.94%	7.77	12.42	23.9%
IGNNK	7.14	10.21	19.83%	7.26	10.80	21.99%	7.11	10.34	20.91%	7.15	10.36	20.85%
STAR	5.66	8.52	14.92%	5.81	8.87	15.38%	5.73	8.85	15.33%	5.65	8.60	14.65%
	PEMS-Bay T=24			PEMS-Bay T=36			PEMS-Bay T=48			PEMS-Bay T=60		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
GCN	5.70	11.42	15.40%	5.77	11.47	15.53%	5.78	11.62	15.47%	5.87	12.17	15.69%
IGNNK	4.11	6.95	10.28%	4.32	7.24	10.35%	4.31	6.93	10.17%	4.39	7.10	10.04%
STAR	3.63	6.15	8.47%	3.71	6.32	8.96%	3.60	5.96	8.30%	3.65	6.13	8.56%



(a) (b)

Fig. 8. Spatial-temporal aware recovery capability. (a) The comparison of (non)temporal approaches on RMSE under different lengths of horizons. (b) The comparison of (non)spatial approaches on RMSE under different lengths of horizons. The suffix -S and -T indicates that the corresponding feature extraction block is enabled.



(a) (b)

Fig. 9. Long-term data recovery capability. (a) The change in MAE and RMSE of STAR model under different recovery horizons. (b) the RMSE errors of the STAR model and other baselines under different recovery horizons.

3) Long Range Recovery: It shows that the proposed model success in obtaining the best recovery performance regardless of the changes in the prediction lengths. Furthermore, the performance is stable with the increase in time steps, and thus, the proposed model can be applied for both short-term and long-term imputation.

As shown in Figure Figure 9(a) the change of MAE and RMSE at varied recover lengths, we learn that it changes slowly with time step increase by a large margin. In addition, as depicted in Figure Figure 9(b), the proposed model is compared with baselines and demonstrates that it outperforms all methods, and added to the fact that it is not sensitive to the length, the imputation relies more on local features than the global one.

G. Ablation Study

To examine the effect of the key components that contribute to the improved outcomes of STAR, we conduct experiments on two traffic datasets. Here, we concentrate on the three kinds of factors: spatial block, temporal block, and external attention. For each factor, a new model is built by removing corresponding blocks, and we named the variants of STAR as follows:

- **w/o EA:** This is STAR without adaptive weighted external attention modules to capture semantic similarity. The graph convolution layer is replaced with diffusion convolution.
- **w/o T:** This is STAR without temporal feature extraction branch before graph convolution layers.

TABLE VI
ABLATION STUDY ON DIFFERENT MODULES

	METR/Degradation		PEMS/Degradation	
Full	5.80	-	3.58	-
w/o EA	6.33	-9.13%	3.68	-2.79%
w/o T	6.06	-4.48%	3.63	-1.39%
w/o S	6.22	-7.24%	3.74	-4.46%

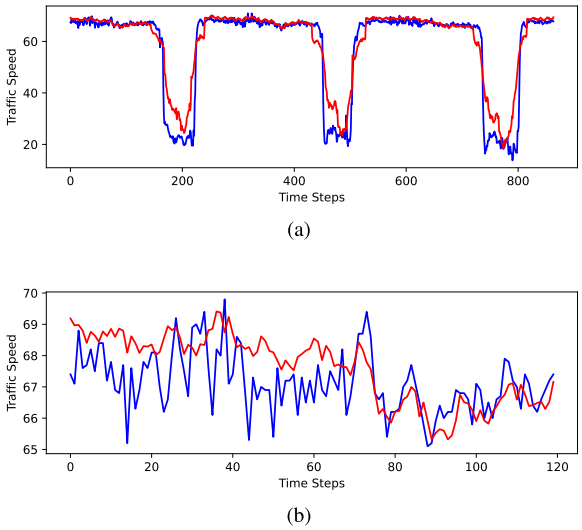


Fig. 10. Data recovery for 24 steps.

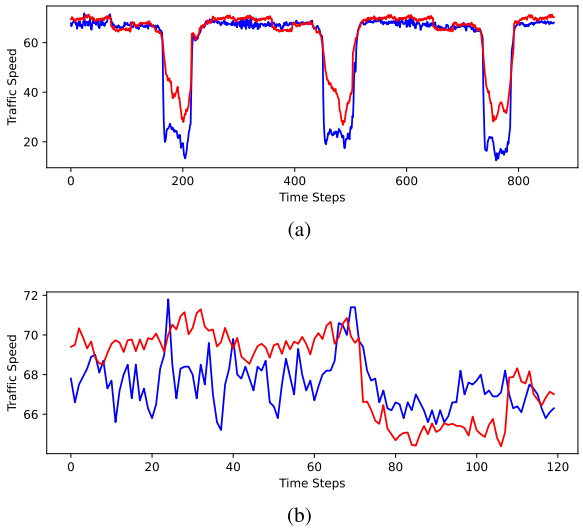


Fig. 11. Data recovery for 36 steps.

- **w/o S:** This is STAR without spatial feature extraction branch.

We assess the performance with the early-stopping strategy to prevent overfitting and present experimental results in Table VI. The introduction of external attention modules significantly improves the performance by providing global sample-wise attention with trainable fractions. We can see the sharp decrease in the accuracy of semantic feature extraction (**w/o S v.s. w/o EA v.s. Full**), indicating the strong correlation and rich semantic similarity inside traffic sensor data series. One explanation to why attention can improve accuracy is that it learns the similarity between nodes like matrix factorization. By implicitly learning the low-rank property, one can accurately recover missing entries by learning from a similar node. The ablation study on the feature extraction, i.e., spatial feature only (**w/o T**) and temporal feature only (**w/o S**), will degrade the performance, which shows that two branches before graph convolution layers can effectively capture the spatial-temporal features for further data imputation. The improvement indicates that, for data imputation tasks, it is better to embed nodes into spatial-temporal context before diffusion through the adjacent graph. Compare (**w/o S**) with (**w/o T**), we find that spatial features share more importance than temporal features, which demonstrates that the traffic speed may impact more by nearby traffic conditions.

H. Imputation Visualization

To better understand the behavior of the STAR model, we randomly select one sensor on PEMS-Bay and visualize the recovery results at different prediction lengths. The following four figures show the recovered time series and the ground truth values in test set of 2 and 5 hours. The results are shown in Figure 10, Figure 11, Figure 12, and Figure 13:

Through the figures presented above, we can learn that our model successfully captured the periodicity of traffic data. Moreover, our model generally provides comparable results for the data series without a clear trend or periodical pattern.

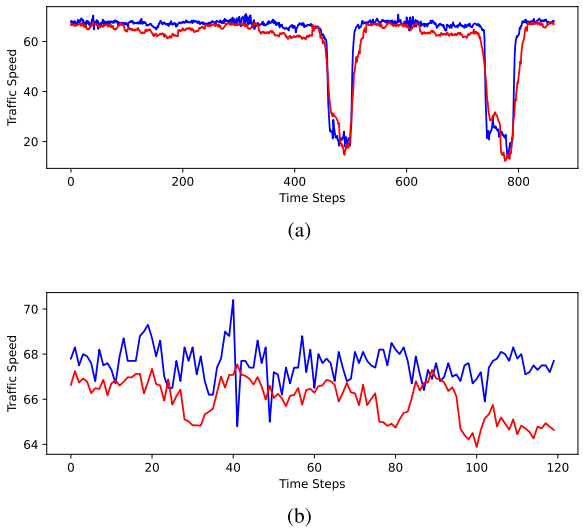


Fig. 12. Data recovery for 48 steps.

Besides, by jointly analyzing four figures, we can learn that the prediction performance has no apparent changes, indicating a long sequences processing capability.

I. Time Series Prediction

As a particular segment type is missing, the time series prediction problem could fit into our data imputation framework if we changed the mask to force the model to impute the missing values at the end of observed windows. With this in mind, we experiment to investigate whether our model can be applied to forecast tasks. We train our model from scratch to predict traffic data using the same setting in [16]. The time series prediction results are presented in Table VII.

According to Table VII, we learn that our model has highly competitive performance in time series prediction though it is designed for data imputation. This feature is due to the power of RG-TCN and GNNs, which effectively extract the temporal patterns and propagate them to nearby sensors. We notice that

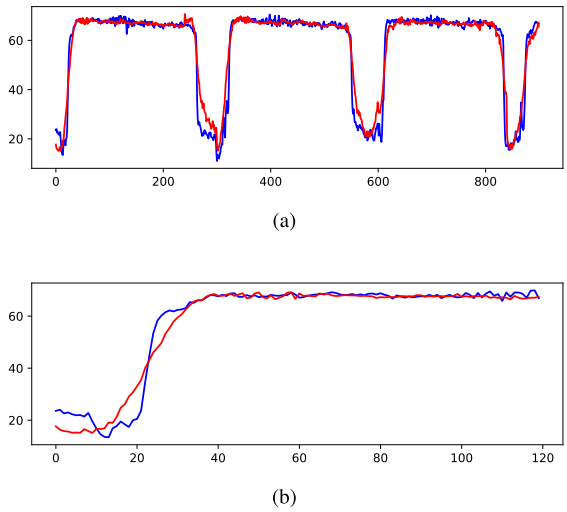


Fig. 13. Data recovery for 60 steps.

TABLE VII

TIME SERIES PREDICTION ACCURACY ON TRAFFIC DATASET

	15/45/60min		
	MAE	RMSE	MAPE
METR-LA			
ARIMA	3.99/5.15/6.9	8.21/10.45/13.23	9.60%/12.70%/17.40%
VAR	4.42/5.41/6.52	7.89/9.13/10.11	10.20%/12.70%/15.80%
FC-LSTM	3.44/3.77/4.37	6.30/7.23/8.69	9.60%/10.90%/13.20%
STGCN	2.88/3.47/4.59	5.74/7.24/9.4	7.62%/9.57%/12.70%
STAR-F	2.81/3.25/3.85	5.02/5.61/6.48	7.07%/8.48%/10.36%
PEMS-Bay			
ARIMA	1.62/2.33/3.38	3.30/4.76/6.50	3.50%/5.40%/8.30%
VAR	1.74/2.32/2.93	3.16/4.25/5.44	3.60%/5.00%/6.50%
FC-LSTM	2.05/2.20/2.37	4.19/4.55/4.96	4.80%/5.20%/5.70%
STGCN	1.36/1.81/2.49	2.96/4.27/5.69	2.90%/4.17%/5.79%
STAR-F	1.32/1.62/2.11	2.39/3.03/3.90	2.72%/3.52%/4.87%

TABLE VIII

TRANSFER PERFORMANCE ON TWO TRAFFIC DATASETS

(Train-Test)	Random/Segment/Blockout		
	MAE	RMSE	MAPE
STAR(M-P)	3.74/5.07/4.28	5.39/6.88/6.83	7.1%/9.6%/9.7%
STAR(P-P)	1.63/2.19/3.52	2.57/3.66/6.09	3.3%/4.8%/8.3%
STAR(P-M)	3.17/4.04/6.37	5.22/6.60/9.52	8.0%/10.9%/16.1%
STAR(M-M)	3.73/4.53/5.71	5.85/6.96/8.50	9.6%/12.1%/14.4%

[1] (A-B) denotes model train on A dataset and test on B dataset.
[2] M and P are short for METR-LA and PEMS-Bay, respectively.

our model has significant advantages in long-range prediction tasks compared with other baselines. Besides, since we forecast the upcoming sensor data for all nodes in one forward computation, our model has lower RMSE and MAPE. It indicates that direct multistep predictions have higher accuracy because of no error accumulation.

J. Transfer Learning

References [3] and [24] reported well-designed GNNs can learn general message passing mechanisms and generalize to a similar dataset. We investigate this phenomenon and report the numerical results in the following table:

We observe from the results obtained that it is possible to train the model on one dataset and directly apply it to another dataset with competitive performance. For example, when we train STAR on METR-LA, it shows a sharp degradation when transferring to PEMS-Bay. In contrast, when we train STAR on PEMS-Bay and test on METR-LA, the transferred model performs even better than the non-transfer model on random missing and segment missing.

This exciting result indicates that the real-world traffic data may share similar spatial-temporal patterns. Furthermore, considering the high similarity between METR-LA and PEMS-Bay, since they collect traffic data every five minutes and provide a distance matrix, pre-trained models and transfer learning have full potential for data-driven traffic analysis.

K. Complexity Analysis

Before we analyze the complexity, we introduce some notation first. For instance, let N denote the number of sensors, T represents the length of the input time series, E denotes the edge in graph G, and d means the hidden model units at each layer.

1) Time Complexity: First, the proposed model receives input at size $N \times T$, which is identical to many other imputation models. Second, we apply RG-TCN as a temporal feature extraction module, and the time complexity is $O(NT)$ (can be viewed as sliding window move over the all input time series). Third, the DiffConv layer propagates the node embeddings in a message-passing manner, thereby, has $O(Ed)$ time complexity. The attention model has $O(Nd)$ time complexity. By adding them up together, we have our model time complexity as $O(NT) + O(Ed) + O(Nd)$.

2) Memory Complexity: Assume that our operation is fully in-place operation. First, the input occupies $O(NT)$ memory. Second, the graph convolution and attention need $O(Nd)$ space to store the immediate results. Therefore, the whole memory complexity is $O(NT) + O(Nd)$.

3) Numerical Results: The parameters of the proposed model occupy 700Kb disk space, and the inference speed on a server with one single NVIDIA K80 GPU is 55ms (average) for 325 nodes with 60-time slots. This significant result shows that the proposed model can be served in an online manner with low latency.

V. CONCLUSION AND FUTURE DIRECTIONS

This article introduces a novel framework for spatial-temporal aware inductive data imputation, namely STAR. The GNNs with an attention-based spatial feature extraction block are enhanced to capture long-range spatial similarity and dilated convolution-based temporal feature extraction. Besides, the proposed model is inductive, which means it can generalize to unseen nodes with retraining. Results obtained from extensive experimentations show that STAR consistently outperforms baseline models on three real-world traffic sensor datasets. Furthermore, analysis of the results demonstrates that the proposed model is insensitive to prediction length, as also its flexibility permits applying it for any data recovery task and model time-varying systems, such as predicting sensor data for moving autonomous cars.

From the significant results and analysis obtained, we foresee some directions as future work to be explored: (1) to extend the proposed model further to support multivariate data imputation, as there are implicit correlations between collected

series that can improve data recovery and decision-making, (2) the high accuracy time series forecast can be explored and then implemented into the proposed model, (3) to develop a unified model to handle all kinds of missing data problems, and lastly, and (4) to optimize the proposed model to be more efficient, meeting the specific requirements of low latency real-time applications.

REFERENCES

[1] W. Liang, D. Zhang, X. Lei, M. Tang, K.-C. Li, and A. Y. Zomaya, "Circuit copyright blockchain: Blockchain-based homomorphic encryption for IP circuit protection," *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 3, pp. 1410–1420, Jul. 2021.

[2] W. Liang, S. Xie, D. Zhang, X. Li, and K.-C. Li, "A mutual security authentication method for RFID-PUF circuit based on deep learning," *ACM Trans. Internet Technol.*, vol. 22, no. 2, pp. 1–20, May 2022.

[3] Y. Wu, D. Zhuang, A. Labbe, and L. Sun, "Inductive graph neural networks for spatiotemporal Kriging," in *Proc. AAAI*, 2021, pp. 4478–4485.

[4] G. Appleby, L. Liu, and L.-P. Liu, "Kriging convolutional networks," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 4, pp. 3187–3194.

[5] M. Brambilla, M. Nicoli, G. Soatti, and F. Deflorio, "Augmenting vehicle localization by cooperative sensing of the driving environment: Insight on data association in urban traffic scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 4, pp. 1646–1663, Apr. 2020.

[6] B. Leblanc, H. Fouchal, and C. de Runz, "Obstacle detection based on cooperative-intelligent transport system data," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jul. 2020, pp. 1–6.

[7] Y. Wu, H. Tan, Z. Jiang, and B. Ran, "ES-CTC: A deep neuroevolution model for cooperative intelligent freeway traffic control," 2019, *arXiv:1905.04083*.

[8] M. Autili, L. Chen, C. Englund, C. Pompilio, and M. Tivoli, "Cooperative intelligent transport systems: Choreography-based urban traffic coordination," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2088–2099, Apr. 2021.

[9] M. A. Javed, S. Zeadally, and E. B. Hamida, "Data analytics for cooperative intelligent transport systems," *Veh. Commun.*, vol. 15, pp. 63–72, Jan. 2019.

[10] W. Liang, Y. Li, J. Xu, Z. Qin, and K.-C. Li, "QoS prediction and adversarial attack protection for distributed services under DLaaS," *IEEE Trans. Comput.*, to be published.

[11] C. Chen, J. Hu, Q. Meng, and Y. Zhang, "Short-time traffic flow prediction with ARIMA-GARCH model," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2011, pp. 607–612.

[12] Y. Tian, K. Zhang, J. Li, X. Lin, and B. Yang, "LSTM-based traffic flow prediction with missing data," *Neurocomputing*, vol. 318, pp. 297–305, Nov. 2018.

[13] H. Lu, Z. Ge, Y. Song, D. Jiang, T. Zhou, and J. Qin, "A temporal-aware LSTM enhanced by loss-switch mechanism for traffic flow forecasting," *Neurocomputing*, vol. 427, pp. 169–178, Feb. 2021.

[14] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. Int. Conf. Learn. Represent.*, 2018.

[15] L. Zhao, Y. Song, C. Zhang, and Y. Liu, "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 9, pp. 3848–3858, Sep. 2020.

[16] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 3634–3640.

[17] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph WaveNet for deep spatial-temporal graph modeling," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019.

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

[19] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 2006.

[20] N. Cressie and C. K. Wikle, *Statistics for Spatio-Temporal Data*. Hoboken, NJ, USA: Wiley, 2015.

[21] J. Yoon, J. Jordon, and M. Schaar, "GAIN: Missing data imputation using generative adversarial nets," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 5689–5698.

[22] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1025–1035.

[23] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. Prasanna, "Graph-SAINT: Graph sampling based inductive learning method," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[24] M. Zhang and Y. Chen, "Inductive matrix completion based on graph neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[25] T. Zhou, H. Shan, A. Banerjee, and G. Sapiro, "Kernelized probabilistic matrix factorization: Exploiting graphs and side information," in *Proc. SIAM Int. Conf. Data Mining*, Apr. 2012, pp. 403–414.

[26] J. Strahl, J. Peltonen, H. Mamitsuka, and S. Kaski, "Scalable probabilistic matrix factorization with graph-based priors," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 4, pp. 5851–5858.

[27] D. Deng, C. Shahabi, U. Demiryurek, L. Zhu, R. Yu, and Y. Liu, "Latent space model for road networks to predict time-varying traffic," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1525–1534.

[28] K. Takeuchi, H. Kashima, and N. Ueda, "Autoregressive tensor factorization for spatio-temporal predictions," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2017, pp. 1105–1110.

[29] M. T. Bahadori, Q. R. Yu, and Y. Liu, "Fast multivariate spatio-temporal analysis via low rank tensor learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 3491–3499.

[30] A. B. Said and A. Erradi, "Spatiotemporal tensor completion for improved urban traffic imputation," *IEEE Trans. Intell. Transp. Syst.*, early access, Mar. 11, 2021, doi: 10.1109/TITS.2021.3062999.

[31] X. Chen, Z. He, Y. Chen, Y. Lu, and J. Wang, "Missing traffic data imputation and pattern discovery with a Bayesian augmented tensor factorization model," *Transp. Res. C, Emerg. Technol.*, vol. 104, pp. 66–77, Jul. 2019.

[32] X. Chen, M. Lei, N. Saunier, and L. Sun, "Low-rank autoregressive tensor completion for spatiotemporal traffic data imputation," *IEEE Trans. Intell. Transp. Syst.*, early access, Sep. 27, 2021, doi: 10.1109/TITS.2021.3113608.

[33] X. Chen, Y. Chen, N. Saunier, and L. Sun, "Scalable low-rank tensor learning for spatiotemporal traffic data imputation," *Transp. Res. C, Emerg. Technol.*, vol. 129, Aug. 2021, Art. no. 103226.

[34] S. Bai, J. Zico Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*.

[35] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 933–941.

[36] M.-H. Guo, Z.-N. Liu, T.-J. Mu, and S.-M. Hu, "Beyond self-attention: External attention using two linear layers for visual tasks," 2021, *arXiv:2105.02358*.

[37] M. Xu *et al.*, "Spatial-temporal transformer networks for traffic flow forecasting," 2020, *arXiv:2001.02908*.

[38] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 17804–17815.

[39] 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 *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 753–763.

[40] A. L. Maas *et al.*, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, 2013, vol. 30, no. 1, p. 3.

[41] J. Lei Ba, J. Ryan Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*.

[42] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015.

Wei Liang received the Ph.D. degree in computer science and technology from Hunan University in 2013. He was a Post-Doctoral Scholar at Lehigh University from 2014 to 2016. He is currently a Professor and the Dean of the School of Computer Science and Engineering, Hunan University of Science and Technology, China. He has authored or coauthored more than 140 journal/conference papers, such as IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING, IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS, and IEEE INTERNET OF THINGS JOURNAL. His research interests include blockchain security technology, networks security protection, embedded system and hardware IP protection, fog computing, and security management in wireless sensor networks (WSN).

Yuhui Li is currently a Graduate Student at Hunan University, China. He has published several high-quality peer-reviewed papers on top journals and conferences, including IEEE TRANSACTIONS ON COMPUTERS and IEEE Conference on Multimedia Expo. His research interests include intelligent transportation systems, service computing, blockchain security, and deep learning.

Kun Xie (Member, IEEE) received the Ph.D. degree in computer application from Hunan University, China, in 2007. She is currently a Professor with Hunan University and the Peng Cheng Laboratory, China. She has published over 60 articles in major journals and conference proceedings, including the IEEE/ACM TRANSACTIONS ON NETWORKING, IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE TRANSACTIONS ON SERVICES COMPUTING, SIGMOD, INFOCOM, ICDCS, SECON, DSN, and IWQoS. Her research interests include network measurement, network security, big data, and AI.

Dafang Zhang received the Ph.D. degree in applied mathematics from Hunan University, China, in 1997. He was a Visiting Fellow with Regina University, Canada, from 2002 to 2003; and a Senior Visiting Fellow with Michigan State University, USA, in 2013. He is currently a Professor at the College of Computer Science and Electronic Engineering, Hunan University. He has authored or coauthored more than 230 journal/conference papers and is the principal investigator (PI) for more than 30 large-scale scientific projects. His research interests include dependable systems/networks, network security, network measurement, hardware security, and IP protection.

Kuan-Ching Li (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from the University of São Paulo (USP), Brazil, in 2001. He has published more than 380 scientific papers and articles. He is the coauthor or a co-editor of more than 30 books published by Taylor & Francis, Springer, and McGraw-Hill. His research interests include parallel and distributed computing, big data, and emerging technologies. He is a fellow of IET and a member of AAAS. Additionally, he has been actively involved in many major conferences and workshops as the program/general/steering conference chairperson positions and has organized numerous conferences and workshops. He is the Editor-in-Chief of *Connection Science* and also serves at leading positions for several scientific journals.

Alireza Souri (Senior Member, IEEE) received the Ph.D. degree in computer engineering from the Science and Research Branch, Islamic Azad University, Iran, in 2018. He is currently an Assistant Professor and a Researcher at Halic University, Istanbul, Turkey. He was also recognized by the Iran's National Elites Foundation and awarded as the National Young Elite in 2018, 2019, and 2020. He has authored/coauthored more than 80 scientific articles and conference papers in high-ranked journals and an associate editor and a guest editor for several well-known scientific journals. His research interests include formal verification, model checking, fog and cloud computing, the Internet of Things (IoT), data mining, and wireless networks.

Keqin Li (Fellow, IEEE) is currently a SUNY Distinguished Professor of computer science with the State University of New York. He is also a National Distinguished Professor with Hunan University, China. He has authored or coauthored more than 780 journal articles, book chapters, and refereed conference papers, and has received several best paper awards. He holds over 60 patents announced or authorized by the Chinese National Intellectual Property Administration. His current research interests include cloud computing, fog computing and mobile edge computing, energy-efficient computing and communication, embedded systems and cyber-physical systems, heterogeneous computing systems, big data computing, high-performance computing, CPU-GPU hybrid and cooperative computing, computer architectures and systems, computer networking, machine learning, and intelligent and soft computing. He has chaired many international conferences. He is an Associate Editor of the *ACM Computing Surveys* and the *CCF Transactions on High Performance Computing*. He has served on the editorial boards for the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON CLOUD COMPUTING, the IEEE TRANSACTIONS ON SERVICES COMPUTING, and the IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING.