Heterogeneous Spatio-Temporal Graph Convolution Network for Traffic Forecasting with Missing Values

Weida Zhong*, Qiuling Suo*, Xiaowei Jia†, Aidong Zhang‡, Lu Su§¶
*State University of New York at Buffalo, †University of Pittsburgh, ‡University of Virginia, §Purdue University
Email: {weidazho, qiulings}@buffalo.edu, xiaowei@pitt.edu, aidong@virginia.edu, lusu@purdue.edu

***Abstract*—Accurate traffic prediction is indispensable for intelligent traffic management. The availability of large-scale road sensing data collected by connected wireless sensors and mobile devices have provided unrealized potential for traffic prediction. However, sensory data is often incomplete due to various factors in the process of data acquisition and transmission. The missingness of traffic data brings a key challenge to the traffic prediction task since the state-of-the-art ML-based traffic prediction models (e.g., Graph Convolutional Networks (GCN)) often rely on spatial and temporal completion of the data. Moreover, existing GCN-based methods usually build a static graph based on geographical distances and are limited in their ability to capture the time-evolving relationships amongst road segments. In this paper, we develop a heterogeneous spatio-temporal prediction framework for traffic prediction using incomplete historical data. In the framework, we build multiple graphs to explicitly model the dynamic correlations among road segments from both geographical and historical aspects, and employ recurrent neural networks to capture temporal correlations for each road segment. We impute missing values in a recurrent process, which is seamlessly embedded in the prediction framework so they can be jointly trained. The proposed framework is evaluated on a public dataset of static sensors and a private dataset collected by our roving sensor system. Experimental results show the effectiveness of the proposed framework compared to state-of-the-art methods, and indicate the potential to be deployed into real-world traffic prediction systems.**

***Index Terms*—Traffic prediction, graph neural network, data imputation**

I. Introduction

Recent years have witnessed increasing stresses on traffic systems as a result of the acceleration of urbanization and the growing human population and numbers of vehicles. Intelligent Transportation System (ITS) [1] aims to improve traffic management by leveraging new data-driven and coordinated services and techniques to guide users in the transport networks. Traffic prediction, whose goal is to predict future traffic conditions (e.g., traffic volume or travel time) of road networks using historical observations, is a fundamental task towards building ITS. Accurate traffic prediction is essential for delivering key transportation services, such as traffic volume control, routine schedule and congestion alleviation.

Nowadays, given the massive amount of traffic data generated and collected by connected wireless sensors and mobile devices, there is growing interest in building data-driven methods for traffic prediction. Recent studies commonly formulate traffic prediction as a spatio-temporal prediction problem. In particular, the state-of-the-art deep learning approaches, e.g., Convolutional Neural Networks (CNN) and Graph Convolutional Network (GCN), have shown much promise in modeling the spatio correlations based on the geographic distance and road connectivity [2]. For example, GCN-based methods have been shown to achieve superior performance over traditional empirical models on several traffic prediction tasks due to its ability to extract complex non-linear dependencies amongst irregular road networks via the graph-structured modeling [3]–[6]. On the other hand, Long Short-Term Memory (LSTM) and temporal convolution have been widely used to capture temporal correlations. Although these techniques have shown some success in isolated studies and relatively simple datasets, they cannot be directly deployed in real-world traffic management systems due to the following challenges:

Missing data over space and time: In modern society, traffic data are mainly collected from two types of sensors: static sensors (e.g., loop detector [3], [5], [7]), and roving sensors (e.g., GPS device on vehicles [8]–[10]). For static sensors, missing data is inevitable due to detector malfunctions, communication errors and transmission failures [11]. For roving sensors, data is often characterized by irregular/non-uniform samples, and the amount of historical data may vary across different segments of a road network [10], resulting in temporal irregularity and spatial sparsity. The existence of missing data from both sensors can negatively impact the performance of aforementioned traffic prediction models as they were originally designed to learn spatio-temporal correlations only from complete data profiles over a continuous time period. Although they can simply remove or fix incomplete samples (e.g., filling zeros or mean values), the prediction performance could drop dramatically with high missing rate. Addressing this challenge requires the development of novel mechanism to impute missing data that recovers spatial and temporal trajectories that are close to the reality.

Dynamic spatial correlations: Most existing GCN-based methods often build static graph structures according to geographic distances or external prior knowledge such as functional similarity and road connectivity [4], [9]. These methods maintain a fixed graph structure over time even under different circumstances. However, the constructed graph structures may not be suitable for predicting future events, as the spatial correlations of traffic data could change over time. For example, it is common to see heavy traffic flow from one road segment to another during the morning peak

¶ Lu Su is the corresponding author.

hours but much less traffic during other time, which shows the non-stationarity of spatial correlations between these two road segments. Hence, relying purely on a single fixed graph, even with some external knowledge, may not be not sufficient to model the dynamic spatial correlations among road segments.

To tackle the challenges, we propose a novel framework: **R**ecurrent **I**mputation based **H**eterogeneous **G**raph **C**onvolution **N**etwork (RIHGCN) for traffic prediction. We handle missing values through a bi-directional recurrent imputation process, in which missing values of different road segments at each timestamp are predicted using spatial and temporal correlations learned from historical data. We then combine imputed and observed values into a "complement" vector and use it for predicting values at the next timestamp. It is noteworthy that imputed values are treated as trainable variables in this computational flow, i.e., they can be updated during the back-propagation of the entire model. This training strategy enables refining the imputation process and prediction process at the same time, and thus prevents imputation errors from impacting the final prediction performance.

Furthermore, to capture dynamic spatial correlations, we build multiple GCNs based on various distance measurements, including geographic distances and historical distances measured by morning peak, evening peak and weekly periodic, etc. The idea is to predict traffic conditions for each road segment by leveraging the information from other road segments with similar historical patterns in each time interval, e.g., morning peak, evening peak. In particular, we create both a static graph based on geographic distances and multiple evolving graphs based on similarities of traffic conditions in each time interval. By using such heterogeneous graphs, our method enables learning more complex and non-stationary spatial relationships amongst road segments.

We evaluate our methods in two real-world traffic datasets. Our experiments show the superiority of our proposed method over existing approaches in both traffic prediction and missing data imputation tasks. Also, we study sensitivity of model performance in response to a varying number of heterogeneous graphs and different values of model hyper-parameters.

Our main contributions can be summarized as follows,

- We design a novel traffic prediction framework to predict traffic conditions and impute missing values simultaneously. This can effectively address the limitation of traditional imputation methods where potential errors from imputation may negatively impact the final prediction.
- We propose a heterogeneous graph structure, which consists of multiple sub-graphs built on both geographic distances and various temporal similarities, to represent the evolving spatial correlations among road segments. Through learning a GCN for each sub-graph and aggregate the learned representations for each road segment, we can better model the non-stationary underlying spatial relationships.
- We impute missing values in a bi-directional recurrent process, and capture complex spatio-temporal correlations by combining GCN and LSTM. The missing values are involved in the backpropagation process, and get delayed gradients in both forward and backward directions with a consistency constraint, which makes the estimation more accurate.
- We empirically show that the proposed framework can effectively handle missing values and significantly outperform state-of-the-art traffic prediction models on both static and roving sensory datasets. The proposed method will be built into a transportation application system to provide future traffic conditions to users.

II. Related Work

A. Traffic Prediction

Traffic prediction is a fundamental problem for urban management. Early approaches on traffic prediction include statistical methods such as autoregressive moving average (ARIMA) [12], Kalman filter [13] and Gaussian process [14]. Recently, various deep learning methods have been developed to capture complex spatial-temporal correlations for traffic prediction, and have achieved state-of-the-art performance. For capturing temporal correlations in traffic conditions, LSTM has been employed in recent approaches [6], [7], [15]. Gated convolutional networks [3], [16] and attention mechanism [17], [18] are also utilized for extracting temporal features. To model spatial correlations, CNN has been used to capture dependencies in the Euclidean space and aggregate predictions in rectangular regions [15], [19], [20]. Recent work [3], [4], [16], [21]–[23] employ GCN to model network graphs and capture the non-Euclidean spatial correlations, and have shown superior performance than CNN-based methods. In general, recent traffic prediction methods simultaneously employ LSTM or temporal convolution for capturing temporal correlations and CNN or GCN to capture spatial correlations.

When applied to real-world scenarios with missing data, existing methods either ignore the data missing problem, e.g., remove data samples with missingness [24], or use straightforward strategies to fix it, e.g., replace all missing values with 0s [17]. However, when it comes to a high missing ratio, especially for data collected by roving sensors, the prediction performance of these methods could drop dramatically. Therefore, handling missing values effectively is of great importance.

B. Missing Data Imputation

Data sparsity issue ubiquitously exists in various real-world applications. This has been handled in multiple ways, including data processing using simple strategies (e.g., mean padding), probabilistic methods and machine learning-based data imputation by learning patterns from available data [25]. Amongst these methods, the imputation methods have the flexibility and have also shown a lot of promise in a variety of prediction tasks [26]–[28]. Commonly used imputation methods include K-nearest neighbors [29], matrix factorization [30] and Multivariate Imputation by Chained Equation [31]. Recently, due to the capability of modeling temporal dependencies of sequential observations, RNN-based models have been used in time series imputation [32]–[36]. In particular, [32] uses bi-directional RNN combined with cross-sectional feature regression to estimate the missing values, and [36] incorporates adversarial training and memory networks into RNN model. However, these methods only capture temporal correlations of general time series, while ignore spatial correlations among traffic data. To incorporate the spatial structures, methods based on matrix factorization [27], [37], [38] and tensor decomposition [10], [26] have been proposed for urban prediction.

The above imputation methods might be used to first preprocess the incomplete data, which are then fed into traffic prediction methods (discussed in Section II-A). However, such two-step solutions may amplify the errors and bring extra computational efforts. Our method imputes the missing values and performs traffic prediction jointly, without a need for a separate imputation step.

III. Methodology

In this section, we introduce the proposed traffic prediction framework. We start with the problem definition and the overview of the proposed framework, and then provide detailed descriptions about its internal components.

A. Problem Definition

Given traffic data collected from sensors from N locations, the objective is to predict future traffic conditions of the road network. Following previous studies, we represent a road network as a weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$, where $\mathcal{V}$ is the set of nodes representing road segments in the road network with $|\mathcal{V}| = N$, $\mathcal{E}$ is a set of edges indicating the connectivity between nodes, and $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix representing proximities (e.g., road network distances) between nodes. At each timestamp t, traffic measurements from each node i in the traffic network are denoted as $\mathbf{x}_t^i \in \mathbb{R}^D$, where D is the number of measured features, and the value of the d-th feature is $x_t^{i,d}$. Here $\mathbf{X}_t = [\mathbf{x}_t^1, \mathbf{x}_t^2, \cdots, \mathbf{x}_t^N]^\top \in \mathbb{R}^{N \times D}$ denotes measured features of all the nodes in the road network at time t, and $\mathcal{X} = [\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_T]^\top \in \mathbb{R}^{N \times D \times T}$ denotes collected features over all the T timestamps. Since $\mathcal{X}$ carries missing values, we introduce a masking tensor $\mathcal{M} \in \mathbb{R}^{N \times D \times T}$ to track the position of missing data in $\mathcal{X}$: $m_t^{i,d}$ is set to 0 if the feature $x_t^{i,d}$ (i.e., the d^{th} feature of node i at time t) is missing; otherwise, $m_t^{i,d}$ is set to 1. Figure 1 illustrates the structure of traffic data.

The traffic prediction problem can be defined as follows: Given historical traffic measurements $\mathcal{X}$ of all nodes in the traffic network over past T timestamps, we aim to learn a model to predict traffic conditions $\mathcal{Y} = [\mathbf{Y}_{T+1}, \mathbf{Y}_{T+2}, \cdots, \mathbf{Y}_{T+T'}]^\top \in \mathbb{R}^{N \times D' \times T'}$ of next T' timestamps for all the nodes.

B. Overall Framework

Overview of the proposed framework is shown in Figure 2. The framework captures spatio-temporal correlations and performs traffic prediction and missing data imputation

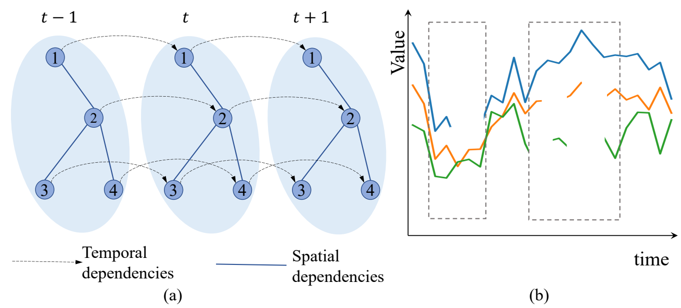


Fig. 1: Traffic data illustration. (a) The spatio-temporal structure of the data. (b) Three variables (e.g., speed at different lanes) are measured on a node across time. Some values are missing as shown inside the dotted rectangle.

simultaneously. In Figure 2 (a), traffic measurements $\mathbf{X}_t$ of all nodes at time t are fed into a heterogeneous GCN (HGCN) block to capture spatial correlations. With obtained node embeddings at different timestamps, we use LSTM to learn temporal correlations of traffic data on each node (i.e., within each road segment) over time. We then concatenate the output vectors of LSTM and HGCN to obtain a hidden state for each node at time t, which captures complex spatio-temporal dependencies of historical measurements in the road network. The hidden states of nodes at time t are used to estimate traffic measurements at the next timestmap $t+1$. All the hidden states across timestamps are aggregated through a fully-connected layer (FC) to perform prediction for traffic condition at future time points $T+1, T+2, \cdots, T+T'$.

Figure 2 (b) illustrates the learning process at one timestamp. At time t, the collected data $\mathbf{X}_t$ carries missing values, i.e., some measurements from certain nodes are missing. Since missing values can hamper model performance, we do not use $\mathbf{X}_t$ as the input of the model directly. Instead, we use a complementary input $\widetilde{\mathbf{X}}_t$ derived by our method to complement the missing values in $\mathbf{X}_t$. Specifically, the HGCN block contains multiple GCNs constructed from geographic structure and spatial dependencies based on historical measurements. The hidden states learned from HGCN and LSTM are used to estimate the values of measurements for each node at time $t+1$. The estimated $\widehat{\mathbf{X}}_{t+1}$ is then combined with $\mathbf{X}_{t+1}$ to obtain $\widetilde{\mathbf{X}}_{t+1}$, which is used as the input at $t+1$ timestamp. This process is performed recurrently to impute the whole sequence of traffic measurements.

C. Basic Graph Convolutional Network

Traffic network is a graph structure in nature. Traffic measurements collected by sensors in different road segments form the features of nodes on the graph, and edges between nodes represent the connectivity of road segments with larger edge weights indicating stronger connections. GCN approaches including spatial-based [39] and spectral-based [40] have been widely used for learning patterns from data with graph structures. Without loss of generality, we adopt the spectral-based GCN in our framework. Note that our method can be

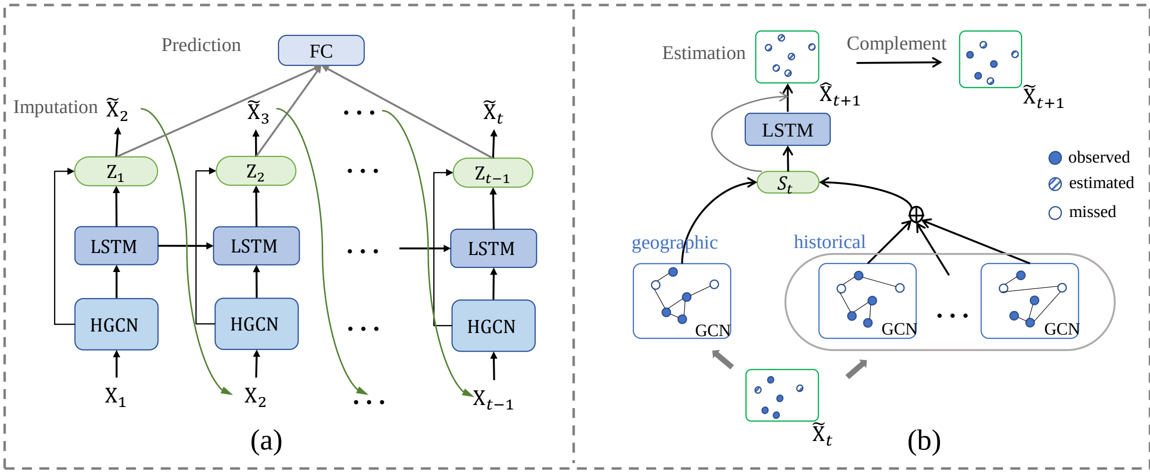


Fig. 2: Framework overview. (a) Overall architecture for traffic prediction, where the output of LSTM at the previous timestamp will be used to fill in the missing values at current timestamp. (b) Basic learning block at one timestamp for learning spatio-temporal correlations and handling missing variables. Heterogeneous graphs are built based upon geographic distance and temporal graphs learned from historical traffic information.

combined with other GCN variants for traffic prediction to enhance the performances.

In a basic GCN approach, the properties of a graph can be obtained by analyzing its corresponding Laplacian matrix. The normalized Laplacian matrix $\mathbf{L} \in \mathbb{R}^{N\times N}$ can be represented as $\mathbf{L} = \mathbf{I}_N - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$, where $\mathbf{I}_N$ is an identity matrix, $\mathbf{A}$ is the adjacent matrix, and $\mathbf{D} \in R^{N\times N}$ is the degree matrix with $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$. Based on spectral graph theory [2], graph convolution can be written as follows,

$$g_\theta *_G x = g_\theta(\mathbf{L})x = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\mathbf{L}})x, \tag{1}$$

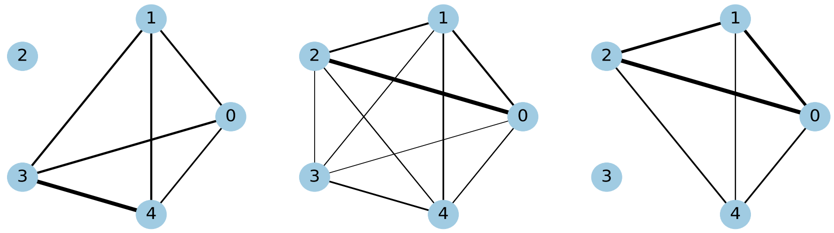
where the vector $\theta \in R^K$ is the polynomial coefficient, $\tilde{\mathbf{L}} = \frac{2}{\lambda_{max}}\mathbf{L} - \mathbf{I}_N$, λ_{max} is the maximum eigenvalue of the Laplacian matrix, and $T_k(\cdot)$ is a Chebyshev polynomial. By using the approximate expansion of Chebyshev polynomial in Eq. (1), the graph convolutional operation extracts information from neighbors within K orders for each node in the graph. In the implementation of GCN, we use generalized graph convolution [3] that is performed on multi-dimensional input vectors.

D. Heterogeneous Graphs

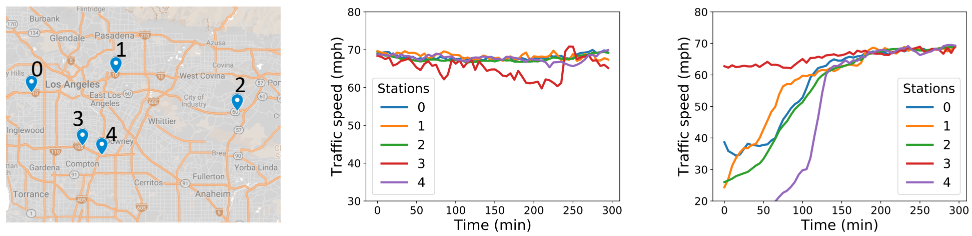
In most existing GCN-based approaches, the graph of road network is constructed solely based on geographic distance, i.e., if two locations are close in the map, the edge connecting them is assigned a large weight. Such a graph structure is static and thus cannot capture the dynamic spatial correlations amongst nodes over time. In Figure 3, we show three graphs constructed from five road segments in the PeMS [41] dataset using different distance measurements. Thicker edges indicate stronger correlations. We can see that although node 2 is far away from other nodes in the geographic graph, it has a very similar time series pattern as node 0. Thus the two nodes are closely connected in temporal graphs. Similarly, although node 3 is close to node 4 by geographic distance, its temporal patterns are quite different from the others, making it less connected with other nodes in temporal graphs. Moreover, the structures of temporal graphs can also vary across different time periods.

To incorporate such heterogeneous information, we propose HGCN which constructs multiple graphs with each one corresponding to a type of spatial correlations. In particular, we divide traffic data in a day into four time intervals: late night, morning, afternoon and evening. We calculate the historical averages of traffic features at the same time period over the past days (e.g., within few months) and obtain a multivariate time series for each interval. We then calculate time series distances between nodes and obtain a temporal graph of the road network for each time interval. Spatial correlations within each temporal graph is learned through a GCN, in which the representation of each node is updated by combining the information from other nodes of similar historical patterns. We then aggregate each node's representations from all the temporal graphs through a weighted summation. Note that we model the daily heterogeneous time periods as an example to show the effectiveness of incorporating dynamic spatial correlations. This method can be easily extended to incorporate more graph structures, e.g., certain time intervals across weeks/months.

It is worthwhile to mention that the graph structures created in our framework are consistent with the typical definition of "heterogeneous graph" - a single graph with different types of edges. Here, we decompose this complex graph into multiple graphs, with each one focusing on a specific type of edge relationship (e.g., road network, similarities of traffic data in a specific time period) so it is clear to show how we learn patterns specific to each type of edges/relationships. However, we can still merge these graphs as a typical heterogeneous



(a) Geographic (b) Temporal (c) Temporal



(d) Road map (e) Late night (f) Evening

Fig. 3: Heterogeneous graphs and the corresponding data. (a) is constructed based on geographic distances between stations in a road map shown in (d). (b) and (c) are constructed based on historical averaged traffic speeds within (e) late night period and (f) evening period, respectively. For (e) and (f), the X-axis is time, and the Y-axis is the averaged traffic speeds.

graph. Our method essentially extracts patterns based on edges of different types in this graph and then merges these learned patterns.

Next, we give more details on the construction procedure of heterogeneous graphs. We build temporal graphs based on different patterns of historical traffic measurements on the road network. In particular, we define a timeline with $\mathcal{T}$ timestamps to track the historical traffic patterns. For example, if we choose a timeline ranging from 00:00 AM to 11:59 PM (24 hours), the timeline will contain 288 timestamps, i.e., $\mathcal{T} = 288$, with 5 minutes per timestamp. Then, for each feature in each node, we calculate the historical average of data at the same timestamps over past days. Therefore, we obtain the historical averaged values for all nodes during the timeline as $\mathbf{H} \in \mathbb{R}^{N \times \mathcal{T} \times D}$, where N is the number of nodes in the graph, and D is the number of measured traffic features.

1) Select the Number of Graphs: We will divide the whole timeline into several intervals and create temporal graphs for each interval. Here we consider the selection of the number of graphs M, i.e., the number of intervals to divide the $\mathcal{T}$ into. Similar to the selection of the number of clusters in the clustering problem, the selection of M depends on the nature of traffic data. If M is too small, then we do not have enough graphs to capture varying spatial correlations across time. If M is too large, the computation overhead will increase but patterns learned from consecutive short intervals can be redundant. We will study the impact of different values of M in the experiment section.

2) Divide the Timeline: After picking up the number of temporal graphs to build, we then split the timeline into M non-overlapping intervals. The length of each interval does not need to be the same. Since we want the graphs to capture varying traffic conditions across time, we need to divide the timeline into intervals in a way such that the total difference between the historical values of each time interval and those of other intervals is maximized. We use $\{t_0, t_1, ..., t_M\}$ to denote the time points to split on, where t_0 is 0, t_M is $\mathcal{T}$, and $t_i < t_{i+1}$. Let $\mathbf{H}_{t_i} \in \mathbb{R}^{N \times (t_i - t_{i-1}) \times D}$ denote the historical measures for all nodes between timestamp t_{i-1} and t_i, $i \in 1, .., M$. Then, we can estimate these split points by solving the following equation,

$$\max_{\{t_1,..,t_M\} \in \{1,..,\mathcal{T}\}} \sum_{i,j} \mathcal{D}(\mathbf{H}_{t_i}, \mathbf{H}_{t_j}), \tag{2}$$

where $\mathcal{D}(\cdot)$ denotes the distance function between series. There are multiple ways to calculate the distance between time series, e.g., Dynamic Time Warping DTW [42], Edit Distance with Real Penalty (ERP) [43], and Longest Common Subsequence (LCSS) [44]. Here, we use DTW to calculate the distance since it can capture the distance between series of variable lengths while does not put too much weight on the difference of amplitude.

One potential problem with Eq. (2) though is that it might produce a trivial solution in which a giant interval dominates other small intervals. To overcome this potential issue, we apply several constraints during the calculation. First, the minimum length of all time intervals must be larger or equal to some threshold, e.g., $\mathcal{T}/(PM)$ where P is a constant. In our settings, this minimum threshold is 1 hour. Second, the maximum length of all time intervals must be smaller or equal to some limit, e.g., $Q\mathcal{T}/M$ where Q is a constant. We set $Q = 2$, i.e., maximum 12 hours. Third, the ratio between the minimum distance among all time intervals and the sum of all distances must be lower or equal to η. η is 10% in the paper.

Finally, the ratio between the length of the longest time interval and the total length of the timeline $\mathcal{T}$ must be lower than a threshold γ, which is set to 50% in our implementation. If we set M to be 4, then after applying these constraints, we can get the output [0 AM, 10 AM), [10 AM, 4 PM), [4 PM, 6 PM), and [6 PM, 0 AM). Such division captures the time periods of early morning, noon, afternoon rush hour, and evening time. A better result could be possible if we form the timeline into a circle so that the first interval does not necessarily start from 00:00 AM, and the last one does not necessarily end at 11:59 PM. We will keep this study as future work.

3) Final Output of HGCN: After we obtain the M separation points of the timeline, we use Eq. (8) to construct the adjacency matrices. For each of the adjacency matrix, we construct a Graph Convolutional Network cell, denoted as GCN_t, following the same process described in Section III-C. For each input sample, we aggregate its outputs of all GCN_ts based on the distance between this sample and the corresponding time interval of each GCN_t cell. This weighted sum along with the output of the GCN constructed based on road geographic distance form the final output of HGCN.

E. Recurrent Imputation

Since traffic data $\mathbf{X}_t$ of road network collected at time t may have missing values, it cannot be used directly as input of a prediction model. Thus we fill in the missing values in $\mathbf{X}_t$ to obtain a complement matrix $\widetilde{\mathbf{X}}_t$ as follows,

$$\widetilde{\mathbf{X}}_t = \mathbf{M}_t \circ \mathbf{X}_t + (1 - \mathbf{M}_t) \circ \widehat{\mathbf{X}}_t, \tag{3}$$

where $\widehat{\mathbf{X}}_t \in \mathbb{R}^{N \times D}$ is the estimated value matrix for traffic measurements at current time t of all nodes, and $\mathbf{M}_t$ is the masking matrix at time t. The matrix $\widehat{\mathbf{X}}_t$ is estimated recurrently using spatial correlations in the graph and temporal correlations in the sequences. At the initial timestamp, we set elements in $\widehat{\mathbf{X}}_0$ to zeros. The complement matrix $\widetilde{\mathbf{X}}_t$ is used as the input of HGCN as below,

$$\mathbf{S}_t = \mathrm{HGCN}(\widetilde{\mathbf{X}}_t), \tag{4}$$

where $\mathbf{S}_t = [\mathbf{s}_t^1, \mathbf{s}_t^2, \cdots, \mathbf{s}_t^N]^\top \in \mathbb{R}^{N \times p}$ is the embedding of N nodes at time t. Here p is the embedding dimension and $\mathbf{s}_t^n \in \mathbb{R}^p$ is the embedding vector for the n-th node. On top of HGCN, we use a recurrent layer to capture temporal correlations of the series for each node n in the graph. At each timestamp t, the recurrent layer combines the information at this timestamp and previous timestamps to jointly generate new data representations. Here we use an LSTM [45] structure to implement the recurrent layer, as follows,

$$\begin{aligned}
\mathbf{f}_t^n &= \sigma(\mathbf{W}_f[\mathbf{s}_t^n; \mathbf{m}_t^n] + \mathbf{U}_f \mathbf{h}_{t-1}^n + \mathbf{b}_f), \\
\mathbf{i}_t^n &= \sigma(\mathbf{W}_i[\mathbf{s}_t^n; \mathbf{m}_t^n] + \mathbf{U}_i \mathbf{h}_{t-1}^n + \mathbf{b}_i), \\
\mathbf{o}_t^n &= \sigma(\mathbf{W}_o[\mathbf{s}_t^n; \mathbf{m}_t^n] + \mathbf{U}_o \mathbf{h}_{t-1}^n + \mathbf{b}_o), \\
\tilde{\mathbf{c}}_t^n &= \tanh(\mathbf{W}_c[\mathbf{s}_t^n; \mathbf{m}_t^n] + \mathbf{U}_c \mathbf{h}_{t-1}^n + \mathbf{b}_c), \\
\mathbf{c}_t^n &= \mathbf{f}_t^n \odot \mathbf{c}_{t-1}^n + \mathbf{i}_t^n \odot \tilde{\mathbf{c}}_t^n, \\
\mathbf{h}_t^n &= \mathbf{o}_t^n \odot \tanh(\mathbf{c}_t^n),
\end{aligned}$$

where $[;]$ is a concatenation operation, $\mathbf{h}_t^n \in \mathbb{R}^q$ is a hidden vector for the n-th node, q is the hidden dimension, and $\mathbf{m}_t^n \in \mathbb{R}^D$ is a masking vector indicating the missing pattern of current collected data $\mathbf{x}_t^n$ of node n. **f**, **i**, and **o** are forget, input, and output gates' activation vectors, respectively, $\odot$ stands for Hadamard product, and $\mathbf{W}_*$, $\mathbf{U}_*$ and $\mathbf{b}_*$ ($* \in \{f, i, o, c\}$) are learnable parameters. For simplicity, we allow all nodes in the graph to share the same LSTM parameters. Therefore, we obtain a hidden state matrix $\mathbf{H}_t = [\mathbf{h}_t^1, \mathbf{h}_t^2, \cdots, \mathbf{h}_t^N]^\top \in \mathbb{R}^{N \times q}$, which captures historical information before the current time t for all the nodes. To mitigate the gradient vanishing problem, we concatenate $\mathbf{S}_t$ and $\mathbf{H}_t$ to obtain an enhanced representation $\mathbf{Z}_t = [\mathbf{S}_t; \mathbf{H}_t] \in \mathbb{R}^{N \times (p+q)}$, which captures complex spatio-temporal dependencies of historical measurements in the road network. Based on $\mathbf{Z}_t$, we can obtain an estimated matrix $\widehat{\mathbf{X}}_{t+1} \in \mathbb{R}^{N \times D}$ for the next timestamp $t+1$ by a linear transformation, as follows,

$$\widehat{\mathbf{X}}_{t+1} = \mathbf{W}_z \mathbf{Z}_t + \mathbf{b}_z, \tag{5}$$

where $\mathbf{W}_z$ and $\mathbf{b}_z$ are learnable parameters. Similar to Eq. (3), we replace missing values in $\mathbf{X}_{t+1}$ with the corresponding estimated values in $\widehat{\mathbf{X}}_{t+1}$ to obtain $\widetilde{\mathbf{X}}_{t+1}$, which is used as input for learning block at time $t+1$. After performing the above process recurrently, we obtain a hidden state tensor $\mathcal{Z} = [\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_T] \in \mathbb{R}^{T \times N \times (p+q)}$ that captures spatio-temporal correlations of all nodes across time.

In the model, $\widehat{\mathbf{X}}_t$ is treated as a trainable variable in the computational flow. This is different from standard LSTM-based imputation models [46] that use zero/mean filled $\mathbf{X}_t$ directly as the input and treat the predicted variable $\widehat{\mathbf{X}}_t$ as a constant during backpropagation. Besides, although temporal convolution [16] and self-attention [17] have been used in replace of LSTM in some traffic prediction methods, they greatly suffer from imperfect spatial and temporal correlations learned from incomplete data trajectories, and also they cannot be easily modified to impute missing data using their original structures. In contrast, by using the recurrent imputation process, our method provides a more accurate version of input ($\widetilde{\mathbf{X}}_t$) to the model. The proposed method also allows delayed error to pass through $\mathbf{X}_t$ to refine estimated values in previous timestamps.

F. Joint Training

In practice, we use bi-directional recurrent process to capture the dependencies from traffic data in past and future timestamps, the hidden state $\mathbf{Z}_t$ is denoted as $\mathbf{Z}_t = [\mathbf{Z}_t^f; \mathbf{Z}_t^b]$ where $\mathbf{Z}_t^f$ and $\mathbf{Z}_t^b$ are obtained from forward and backward directions respectively. We use mean absolute error (MAE) to estimate the imputation loss $\mathcal{L}_m$ as below,

$$\mathcal{L}_m = \frac{1}{T} \sum_{t=1}^{T} \mathbf{M}_t \circ |\mathbf{X}_t - \widehat{\mathbf{X}}_t| + (1 - \mathbf{M}_t) \circ |\widehat{\mathbf{X}}_t^f - \widehat{\mathbf{X}}_t^b|, \tag{6}$$

where $\widehat{\mathbf{X}}_t^f$ and $\widehat{\mathbf{X}}_t^b$ are the estimated values at t from forward and backward directions, respectively, and $\widehat{\mathbf{X}}_t = \frac{1}{2}(\widehat{\mathbf{X}}_t^f + \widehat{\mathbf{X}}_t^b)$. In Eq.(6), the first term measures the error between estimated values and observed values, and the second term enforces the estimation in each step to be consistent in both directions for missing values. Meanwhile, we calculate the traffic prediction error as below,

$$\mathcal{L}_e = \frac{1}{T'} \sum_{t=T+1}^{T+T'} |\mathbf{Y}_t - \mathrm{FC}(\mathcal{Z})|, \tag{7}$$

where $\mathrm{FC}(\cdot)$ is a fully-connected layer. We can concatenate hidden states $\mathbf{Z}_t$ in $\mathcal{Z}$ or use attention mechanism to obtain a weighted sum of hidden states. During the training, we optimize the total loss $\mathcal{L} = \mathcal{L}_e + \lambda \mathcal{L}_m$, where λ is a hyperparameter. Through joint training with imputation as an auxiliary task, we obtain high-quality imputed data which facilitate the learning of spatio-temporal denpendencies of traffic data, and thus could be helpful for the prediction task.

IV. Experiments

In this section, we first describe two real-world datasets that are used in our experiments. Then we propose a few research questions to be answered in our experiments, and introduce our experimental designs, baselines and model implementation details. Finally, we discuss the experimental results to answer the proposed research questions.

A. Datasets

We evaluate the model performances on two real-world datasets: PeMS [41] which is a public dataset collected from static sensors, and Stampede which is collected by our roving sensor system.

1) PeMS: It includes traffic data of California highway that are collected by the Caltrans Performance Measurement System (PeMS) every 30 seconds. The traffic data are aggregated into several different intervals, e.g., 5 minutes and 30 minutes. We collect PeMS traffic speed data of 5 minutes interval in district 07 from January 1, 2020 to April 30, 2020. Four measurements are chosen, including the average speed of all lanes, and lane speeds for the first three lanes.

2) Stampede: We have developed an Android application that can acquire and save real time GPS location at about 1Hz. We installed this application on 15 Android smartphones and deployed the smartphones on 15 shuttles named "Stampede", that run among different locations in the city. The phone is connected to the bus DC power with an adapter so that it can get charged properly. The application starts automatically when the shuttle turns on and shuts down when it loses power. The collected data is first saved onto device's internal storage. When the Stampede is running on campus, the on-board smartphone will connect to the campus WiFi automatically and upload the data to our server.

Here we use collected data of 12 road segments from February 1, 2019 to June 30, 2019. Travel time is collected for each road segment. Road network information, including the number of lanes per direction, number of traffic lights, speed limits, and the GPS location of the center point of each road segment, is used to calculate the adjacency matrix for the geographic graph.

3) Data Preprocessing: Each dataset is divided into training, validation, and test subsets in 7:2:1. The data is normalized using Z-score. The adjacent matrix is calculated as follows,

$$A_{i,j} = \begin{cases} \exp\left(-\frac{d_{i,j}^2}{\sigma^2}\right), & \text{if } \exp\left(-\frac{d_{i,j}^2}{\sigma^2}\right) \geq \epsilon \\ 0, & \text{otherwise} \end{cases}, \tag{8}$$

where $d_{i,j}$ is the distance between node i and j, σ is the standard deviation, and ϵ is the threshold to control the sparsity of the adjacency matrix. ϵ is set to 0.1 in the following experiments. Note that $d_{i,j}$ is calculated differently for different graphs in HGCN.

B. Experimental Settings

1) Evaluation Strategies: We evaluate the proposed method on real-world datasets with the aim to answer the following research questions (RQs):

RQ 1: Does RIHGCN outperform other competitors in the traffic prediction task?
RQ 2: Does it perform data imputation effectively?
RQ 3: Does the heterogeneous graph structure help enhance the learning of traffic patterns?
RQ 4: How does the prediction and imputation performance change w.r.t. the weight of imputation loss in the optimization process?

In particular, for the evaluation of traffic prediction and imputation, we use mean absolute error (MAE) and root mean squared error (RMSE). Smaller MAE/RMSE indicates better performance.

2) Baseline Approaches: We compare our method with various traffic prediction models, including:

- Historical Average (HA): We calculate the average traffic information for each time series, and use it as the predicted value for future timestamps.
- Vector Autoregression (VAR) [47]: It is a statistical model for multivariate time series analysis. Each variable is predicted as a linear function of past lags of itself and the other variables. The number of lags is set to 3.
- ASTGCN [3]: It applies attention and convolution on both spatial and temporal dimensions to capture spatial-temporal dependencies. The Chebyshev polynomial order K is set to 3. For lengths of periodic segments, we set them as $T_d = 12$ and $T_w = 24$ for days and weeks respectively, and we set $T_h = 12$ in accordance with the lookback length of other models.
- Graph WaveNet [24]: It learns an adaptive dependency matrix to capture spatial dependency and stacked temporal convolution to handle long sequences.
- FC-LSTM: We use LSTM to capture temporal correlations for prediction, and aggregates hidden states across time using an FC layer to perform prediction.
- FC-GCN: We use GCN to capture spatial correlations at each timestamp, and aggregate the hidden state of each node for prediction.

TABLE I: Performance on PeMS dataset w.r.t. different missing rates (upper table) and different prediction lengths (lower).

	Methods	20%		40%		60%		80%	
		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Missing Rate	HA	2.2539	4.2306	2.2893	4.2966	2.4295	4.5270	3.1846	5.5498
	VAR	8.1122	32.7502	8.1139	32.8293	8.1120	32.7412	8.1139	32.8293
	FC-LSTM	4.4152	6.5582	4.4472	6.5720	4.6029	6.6994	4.8813	6.9652
	FC-GCN	2.7651	4.3245	2.8936	4.4826	3.0595	4.6905	3.3236	5.0278
	GCN-LSTM	2.2073	3.7735	2.3750	3.9873	2.7792	4.4430	3.2553	5.0007
	ASTGCN	3.6935	5.5691	3.7658	5.6405	3.7687	5.6447	3.9723	5.8757
	Graph WaveNet	2.6757	4.2416	2.8288	4.3083	2.9081	4.4120	3.0958	4.6799
	FC-LSTM-I	4.2205	6.3957	4.2322	6.4278	4.3279	6.5532	4.8064	6.8940
	FC-GCN-I	2.4611	4.0206	2.4475	4.0455	2.7644	4.3866	3.0110	4.6979
	GCN-LSTM-I	2.1607	3.7381	2.2134	3.8232	2.4777	4.1450	3.0127	4.7521
	RIHGCN	**2.0848**	**3.6598**	**2.1698**	**3.7266**	**2.3304**	**3.9483**	**2.8145**	**4.5136**

	Methods	15 min		30 min		45 min		60 min	
		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Prediction Length	HA	2.8647	5.1024	2.9837	5.2618	3.0864	5.4084	3.1846	5.5498
	VAR	5.7164	24.6579	7.0335	28.6984	7.6454	30.8039	8.1139	32.8293
	FC-LSTM	4.6815	6.8133	4.9416	7.0741	4.5118	6.7570	4.8813	6.9652
	FC-GCN	3.0308	4.5916	3.1387	4.7455	3.2126	4.8651	3.3236	5.0278
	GCN-LSTM	2.5142	4.0778	2.8027	4.4303	2.9428	4.6177	3.2553	5.0007
	ASTGCN	3.4943	5.2039	3.9955	5.8913	3.9343	5.8382	3.9723	5.8757
	Graph WaveNet	2.7600	3.8223	2.7981	4.3126	2.9400	4.4727	3.0958	4.6799
	FC-LSTM-I	4.5618	6.6704	4.4642	6.6932	4.3941	6.6118	4.8064	6.8940
	FC-GCN-I	2.4282	3.9395	2.7176	4.3042	2.8971	4.5420	3.0110	4.6979
	GCN-LSTM-I	2.3466	3.8782	2.4873	4.0804	2.5736	4.2324	3.0127	4.7521
	RIHGCN	**2.1436**	**3.6081**	**2.4834**	**4.0520**	**2.5412**	**4.1685**	**2.8145**	**4.5136**

- GCN-LSTM: It combines GCN and LSTM and feeds hidden states through an FC layer to perform prediction.

Since these methods are designed to learn from complete traffic data and do not handle missingness directly, we first fill the missing values with corresponding mean of observed values, which is a commonly used way in existing traffic prediction models [17] to handle missingness, and then perform prediction using these baseline approaches. To evaluate the contribution of different components in the proposed framework, we conduct the following ablation study.

- FC-LSTM-I: It recurrently imputes missing values for each node using LSTM and aggregates the learned hidden states to perform prediction for future timestamps. Considering the bi-directional recurrent imputation process, it is similar to the time series imputation method BRITS [32]. We compare with this method to investigate the contribution of capturing temporal correlations alone.
- FC-GCN-I: It uses GCN to estimate missing values at the next timestamp, and aggregates node embeddings across time to perform prediction on each node. This method utilizes only spatial correlations.
- GCN-LSTM-I: It imputes missing values by combining GCN and LSTM, and then performs prediction using the spatio-temporal representations. It has a similar structure as the proposed model RIHGCN, but only uses geographic graph without temporal graphs.

To evaluate how accurate our method can recover the missed traffic data, we evaluate the imputation performance of our method by comparing with widely-used imputation approaches including last observed (Last), k-nearest neighbors (KNN), matrix factorization (MF) and tensor decomposition (TD) [10].

For all the deep learning based models, we use Adam optimizer with learning rate of 0.001, and batch size is 64. Early stopping is adopted when the validation performance does not improve for 6 epochs.

3) Experimental Details: All methods are implemented using PyTorch 1.15 with Python 3.7. Adam optimizer [48] is used as the optimization method with learning rate of 0.001 and with gradient clipping. Following previous works [7], we use 12 historical timestamps, i.e., 1 hour, to predict the traffic information for the following up to 12 timestamps. Chebyshev polynomial order K is set to 3. LSTM hidden layer size is 128. The number of GCN filters is 64.

C. Experimental Results

Here we will discuss the experimental results to answer to four RQs raised in Section IV-B1.

*1) Prediction Performance (**RQ1**):* We evaluate model performances on traffic prediction with respect to missing rates and prediction lengths. The results on PeMS dataset is shown in Table I. We compare results under different missing rates, i.e., 20%, 40%, 60% and 80%, which indicate the percentage of values that have been randomly dropped in historical data. The prediction length is 60 min, i.e., 12 timestamps. From the

TABLE II: Performance on Stampede dataset w.r.t. different prediction lengths.

	Methods	15 min		30 min		45 min		60 min	
		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Prediction Length	HA	28.4851	38.0368	28.5613	38.1846	28.6079	38.2897	28.6077	38.2821
	VAR	29.0215	40.6851	29.1390	41.3184	29.5232	41.4406	29.6419	41.5024
	FC-LSTM	25.9037	34.5826	25.9342	34.6202	25.9741	34.7271	26.1448	34.9650
	FC-GCN	25.8750	34.5702	25.9442	34.6199	25.8948	34.7211	26.0231	34.9132
	GCN-LSTM	25.8364	34.5307	25.9016	34.5966	25.8921	34.6834	25.9502	34.8974
	ASTGCN	26.0712	34.6109	26.1855	35.0434	26.0947	34.9150	26.2375	35.0060
	Graph WaveNet	25.8247	34.5063	25.9493	34.5610	25.8991	34.6173	26.0109	35.5515
	FC-LSTM-I	25.8809	34.5363	25.9071	34.6099	25.9337	34.6602	26.0251	34.9601
	FC-GCN-I	25.8280	34.5026	25.8975	34.5868	25.9019	34.6526	26.0169	34.8992
	GCN-LSTM-I	25.7727	34.4781	25.8837	34.5347	25.8785	34.5933	25.9613	34.8101
	RIHGCN	**25.7083**	**34.4347**	**25.8327**	**34.4708**	**25.8581**	**34.5027**	**25.9220**	**34.6614**

table we can observe that with the increasing of the missing rate, the performance of all the methods drops. Methods with recurrent imputation process (e.g., FC-LSTM-I) generally has better performances compared with the corresponding models without handling imputation during training (e.g., FC-LSTM). so they cannot perform well on data with missingness. This is because the state-of-the-art traffic prediction approaches ASTGCN and Graph WaveNet do not perform well on our datasets. This is because they are originally designed for complete datasets and thus cannot effectively handle missing data. The proposed model achieves the best performances under almost all cases, especially when the missing rate is high. This is due to the fact that our method can estimate missing values accurately and extract predictive features from historical data with missingness.

In the bottom of Table I, we show the prediction performance under different prediction lengths where the missing rate is fixed to 80%. We can observe the similar performance upgrade from basic prediction models to their imputation-enhanced variants, e.g., FC-LSTM vs. FC-LSTM-I, and FC-GCN vs. FC-GCN-I. By comparing with the reduced versions of our method (i.e., FC-LSTM-I, FC-GCN-I, and GCN-LSTM-I), we observe that GCN-LSTM-I performs better than using FC-GCN-I or FC-LSTM-I alone, which indicates the effectiveness of spatio-temporal models for traffic prediction. By incorporating our proposed HGCN instead of GCN, RIHGCN further improves the performance over GCN-LSTM-I.

The prediction performance on Stampede dataset is listed in Table II. Due to the high missing rate which is the common characteristic for roving sensor collected data as a result of limited number of sensors, we only study the performance with respect to different prediction lengths. The prediction on the Stampede dataset is more challenging due to the higher missing rate and more variability in local traffic conditions. Still, the performances of our method are stable and competitive compared with state-of-the-art methods.

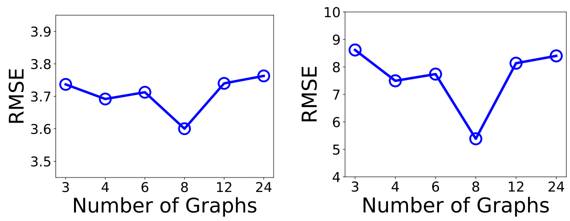
*2) Imputation Performance (**RQ2**):* Here we evaluate the imputation performance under 40% and 80% missing rate. We also randomly remove 30% of the observed entries and use them as the groundtruth for evaluation. Comparison results between imputated and groundtruth values are listed in Table III. For our methods, since we focus on the prediction task, we report the imputation errors when the best prediction performance is achieved, rather than reporting the best imputation performance directly.

We can see that our methods estimate missing values more accurately compared to other widely-used imputation methods. The main limitation of these traditional imputation approaches (e.g., Last, KNN, MF, and TD) is that they cannot fully capture the complex nonlinear and non-stationary data dependencies over space and time. RIHGCN performs much better than other methods when we have a lower missing rate (40%). As we increase the missing rate for training data, most methods have increased imputation errors due to the reduction in available observation data. The GCN model (FC-GCN-I) performs the best given that it has simpler structure (i.e., having small number of model parameters) while also preserving dependencies across road segments. Our method RIHGCN can still achieve comparable performance the best model in this case.

TABLE III: Imputation performance on PeMS dataset.

Methods	40%		80%	
	MAE	RMSE	MAE	RMSE
Last	6.9841	8.9942	6.9811	9.0001
KNN	5.1798	7.7112	6.5500	9.5230
MF	6.4746	8.3086	6.7054	9.0430
TD	4.7228	6.8110	7.4593	9.9909
FC-LSTM-I	6.2962	8.2106	6.8382	8.8333
FC-GCN-I	5.5202	8.0193	**5.8595**	**8.4615**
GCN-LSTM-I	6.3170	8.8719	6.8751	9.5835
RIHGCN	**4.2804**	**6.2702**	5.9915	8.5884

*3) Impact of the Number of Graphs (**RQ3**):* In this part, we study the effectiveness of building multiple heterogeneous graphs for learning representations. In particular, we measure the performance of prediction and imputation using different numbers of graphs. A larger number of graphs indicates finer-level time intervals, e.g., three graphs can capture the variation of temporal patterns for every 8 hours while 24 graphs can capture the change across every hour (if all time intervals share the same length). We show the prediction performance and imputation performance in Figure 4. Here we set the missing rate as 40% and the prediction length as 12. From the figures we can see that both prediction and imputation achieve the best performance when the number of graphs is 8. When the number of graphs is too small, the time interval is too long so that it does not capture the variability of traffic conditions in a day effectively. When the number of graphs is too large, we create too many intervals resulting in much redundancy between consecutive intervals. This also brings additional complexity and computational cost to our model.



(a) Prediction (b) Imputation

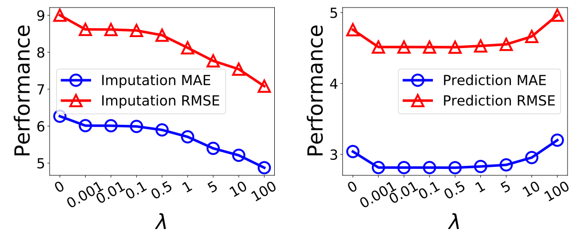
Fig. 4: Performance of (a) traffic prediction and (b) data imputation w.r.t. the number of graphs on PeMS dataset.

*4) Parameter Study (**RQ4**):* To evaluate the effect of λ which controls the weight of imputation loss, we report both the imputation and prediction performance with respect to λ. Larger λ indicates more penalty on imputation. From Figure 5, we observe that the imputation performance continues to increase with the increasing value of λ. This confirms that the model can better impute the missing data as we force the model to pay more attention on the imputation loss. On the other hand, our proposed method has good prediction performance when $\lambda \in (0.001, 5)$. However, the prediction performance decreases when λ is very small (<0.001) or very large (>5). This is due to the fact that smaller penalty on imputation could result in a large error in estimating the missed historical data, and this error would negatively affect the prediction task. A large λ may cause overfitting of imputation, i.e., focusing too much on details of historical data but lacking the ability to capture predictive signals.

V. Conclusion

In this paper, we propose RIHGCN for traffic prediction with missing values. Due to the nature of traffic sensors, the collected traffic data inevitably carry missing values, and the missingness hampers the performance of various state-of-the-art traffic prediction methods. We effectively impute missing values by utilizing the spatio-temporal correlations among nodes in the road network through a recurrent imputation process, and propose a heterogeneous graph structure to capture dynamic spatial correlations among nodes.

Our contributions mainly lie in two aspects: 1) Different from standard GCN model which uses a static graph structure (e.g., created using geographic features), we propose to create multiple graphs with different types of edges using



(a) Imputation (b) Prediction

Fig. 5: Performance of (a) data imputation and (b) prediction w.r.t. the weight of imputation loss on PeMS dataset.

the similarities at different time intervals. These graphs can help better capture spatial correlations that change over time in traffic data. 2) We integrate the missing data imputation and traffic prediction in a unified framework. Indeed, imputing missing values in traffic prediction is a classic problem, and we've listed related work on this problem in Section II-B. However, most methods focus on imputation purely, and they cannot be easily integrated with the downstream task, i.e., traffic prediction via an advanced spatio-temporal network, in a seamless fashion. The errors resulting from the imputation can be accumulated to the downstream tasks. In contrast, our proposed unified framework optimizes the two objectives (data imputation and traffic prediction) simultaneously via capturing spatio-temporal correlations, and thus alleviates this issue.

Experimental results show that our method outperforms existing methods by a considerable margin in both prediction and imputation tasks. Different from existing traffic prediction methods, we show the superiority of the proposed unified framework that conducts imputation and prediction simultaneously in a complementary fashion. We anticipate this work to provide solutions to a broad class of spatio-temporal prediction problems with incomplete data, e.g., air quality prediction with data collected in different locations of a city.

Acknowledgment

This work was supported in part by the US National Science Foundation under Grants CNS-1652503 and CNS-1737590. The views and conclusions in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of NSF.

References

[1] L. Li, X. Qu, J. Zhang, Y. Wang, and B. Ran, "Traffic speed prediction for intelligent transportation system based on a deep feature fusion model," *Journal of Intelligent Transportation Systems*, vol. 23, no. 6, pp. 605–616, 2019.

[2] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[3] 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, 2019, pp. 922–929.

[4] X. Geng, Y. Li, L. Wang, L. Zhang, Q. Yang, J. Ye, and Y. Liu, "Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 3656–3663.

[5] Q. Zhang, J. Chang, G. Meng, S. Xiang, and C. Pan, "Spatio-temporal graph structure learning for traffic forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 1177–1185.

[6] W. Chen, L. Chen, Y. Xie, W. Cao, Y. Gao, and X. Feng, "Multi-range attentive bicomponent graph convolutional network for traffic forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

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

[8] W. Zhong, Q. Suo, F. Ma, Y. Hou, A. Gupta, C. Qiao, and L. Su, "A reliability-aware vehicular crowdsensing system for pothole profiling," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 4, pp. 1–26, 2019.

[9] 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.

[10] Y. Wang, Y. Zheng, and Y. Xue, "Travel time estimation of a path using sparse trajectories," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 25–34.

[11] L. Qu, L. Li, Y. Zhang, and J. Hu, "Ppca-based missing data imputation for traffic flow volume: A systematical approach," *IEEE Transactions on intelligent transportation systems*, vol. 10, no. 3, pp. 512–522, 2009.

[12] M.-C. Tan, S. C. Wong, J.-M. Xu, Z.-R. Guan, and P. Zhang, "An aggregation approach to short-term traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 1, pp. 60–69, 2009.

[13] Y. Wang, M. Papageorgiou, and A. Messmer, "A real-time freeway network traffic surveillance tool," *IEEE Transactions on control systems technology*, vol. 14, no. 1, pp. 18–32, 2005.

[14] J. Chen, K. H. Low, Y. Yao, and P. Jaillet, "Gaussian process decentralized data fusion and active sensing for spatiotemporal traffic modeling and prediction in mobility-on-demand systems," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 901–921, 2015.

[15] H. Yao, X. Tang, H. Wei, G. Zheng, and Z. Li, "Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 5668–5675.

[16] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," *arXiv preprint arXiv:1709.04875*, 2017.

[17] 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*, vol. 34, no. 01, 2020, pp. 1234–1241.

[18] X. Jia, S. Li, A. Khandelwal, G. Nayak, A. Karpatne, and V. Kumar, "Spatial context-aware networks for mining temporal discriminative period in land cover detection," in *Proceedings of the 2019 SIAM International Conference on Data Mining*. SIAM, 2019, pp. 513–521.

[19] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017.

[20] Y. Li, K. Fu, Z. Wang, C. Shahabi, J. Ye, and Y. Liu, "Multi-task representation learning for travel time estimation," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1695–1704.

[21] C. Chen, K. Li, S. G. Teo, X. Zou, K. Wang, J. Wang, and Z. Zeng, "Gated residual recurrent graph neural networks for traffic prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 485–492.

[22] Z. Diao, X. Wang, D. Zhang, Y. Liu, K. Xie, and S. He, "Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 890–897.

[23] X. Jia, J. Zwart, J. Sadler, A. Appling, S. Oliver, S. Markstrom, J. Willard, S. Xu, M. Steinbach, J. Read *et al.*, "Physics-guided recurrent graph model for predicting flow and temperature in river networks," in *Proceedings of the 2021 SIAM International Conference on Data Mining*. SIAM, 2021.

[24] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," in *IJCAI*, 2019.

[25] A. N. Baraldi and C. K. Enders, "An introduction to modern missing data analyses," *Journal of school psychology*, vol. 48, no. 1, pp. 5–37, 2010.

[26] A. Baggag, T. Zanouda, S. Abbar, and F. Filali, "Learning spatiotemporal latent factors of traffic via a regularized tensor factorization: Imputing missing values and forecasting," *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[27] T. Han, K. Wada, and T. Oguchi, "Large-scale traffic data imputation using matrix completion on graphs," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 2252–2258.

[28] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Scientific reports*, vol. 8, no. 1, pp. 1–12, 2018.

[29] A. T. Hudak, N. L. Crookston, J. S. Evans, D. E. Hall, and M. J. Falkowski, "Nearest neighbor imputation of species-level, plot-scale forest structure attributes from lidar data," *Remote Sensing of Environment*, pp. 2232–2245, 2008.

[30] M. Morup, D. M. Dunlavy, E. Acar, and T. G. Kolda, "Scalable tensor factorizations with missing data." Sandia National Laboratories, Tech. Rep., 2010.

[31] S. v. Buuren and K. Groothuis-Oudshoorn, "MICE: Multivariate imputation by chained equations in R," *Journal of statistical software*, pp. 1–68, 2010.

[32] W. Cao, D. Wang, J. Li, H. Zhou, L. Li, and Y. Li, "BRITS: Bidirectional recurrent imputation for time series," in *Advances in Neural Information Processing Systems*, 2018, pp. 6775–6785.

[33] Y. Luo, X. Cai, Y. Zhang, J. Xu, and X. Yuan, "Multivariate time series imputation with generative adversarial networks," in *Advances in Neural Information Processing Systems*, 2018.

[34] Y.-J. Kim and M. Chi, "Temporal belief memory: Imputing missing data during rnn training." in *In Proceedings of International Joint Conferences on Artificial Intelligence*, 2018.

[35] Q. Suo, L. Yao, G. Xun, J. Sun, and A. Zhang, "Recurrent imputation for multivariate time series with missing values," in *2019 IEEE International Conference on Healthcare Informatics (ICHI)*. IEEE, 2019, pp. 1–3.

[36] X. Tang, H. Yao, Y. Sun, C. C. Aggarwal, P. Mitra, and S. Wang, "Joint modeling of local and global temporal dynamics for multivariate time series forecasting with missing values." in *AAAI*, 2020, pp. 5956–5963.

[37] Q. Wang, P.-N. Tan, and J. Zhou, "Imputing structured missing values in spatial data with clustered adversarial matrix factorization," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 1284–1289.

[38] X. Chen, Y. Cai, Q. Ye, L. Chen, and Z. Li, "Graph regularized local self-representation for missing value imputation with applications to on-road traffic sensor data," *Neurocomputing*, vol. 303, pp. 47–59, 2018.

[39] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[40] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in neural information processing systems*, 2016, pp. 3844–3852.

[41] C. Chen, K. Petty, A. Skabardonis, P. Varaiya, and Z. Jia, "Freeway performance measurement system: mining loop detector data," *Transportation Research Record*, vol. 1748, no. 1, pp. 96–102, 2001.

[42] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intelligent Data Analysis*, vol. 11, no. 5, pp. 561–580, 2007.

[43] L. Chen and R. Ng, "On the marriage of lp-norms and edit distance," in *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, 2004, pp. 792–803.

[44] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering similar multidimensional trajectories," in *Proceedings 18th international conference on data engineering*. IEEE, 2002, pp. 673–684.

[45] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[46] J. Yoon, W. R. Zame, and M. van der Schaar, "Multi-directional recurrent neural networks: A novel method for estimating missing data," in *International Conference on Machine Learning (ICML) Time Series Workshop*, 2017.

[47] S. R. Chandra and H. Al-Deek, "Predictions of freeway traffic speeds and volumes using vector autoregressive models," *Journal of Intelligent Transportation Systems*, vol. 13, no. 2, pp. 53–72, 2009.

[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.