Context Integrated Relational Spatio-Temporal Resource Forecasting

Hongjie Chen
Department of Computer Science
Virginia Tech
Blacksburg, USA
jeffchan@vt.edu

Ryan A. Rossi
Adobe Research
San Jose, USA
ryrossi@adobe.com

Kanak Mahadik
Adobe Research
San Jose, USA
mahadik@adobe.com

Hoda Eldardiry
Department of Computer Science
Virginia Tech
Blacksburg, USA
hdardiry@vt.edu

***Abstract*—Traditional resource (demand or supply) forecasting models mainly focus on modeling temporal dependency. However, spatio-temporal data include complex non-linear relational and spatial dependencies. In addition, dynamic contextual information also impacts resources. Methods that consider context assume that the impact of context on resources is fixed, which is not realistic. For example, in a bicycle-sharing system, bike supply in stations is affected by the weather, and that effect changes over time. We propose a novel graph-based context integrated relational model, *Context Integrated Graph Neural Network* (CIGNN), which models temporal, relational, spatial, and dynamic contextual dependencies for multi-step ahead resource forecasting. We define a resource graph, where nodes represent locations with associated resource time-series, and context graphs (one for each type of context), where nodes represent locations with associated contextual time-series. Assuming that various contexts have dynamic impact on resources, our proposed CIGNN model employs a novel fusion mechanism that jointly learns from multiple contextual time-series. To the best of our knowledge, CIGNN is the first approach that integrates dynamic contextual information using graph neural networks for resource forecasting. Empirical results on two real-world datasets demonstrate that CIGNN consistently outperforms state-of-the-art approaches.**

***Index Terms*—Demand and supply forecasting, Graph Neural Network, Spatio-temporal time-series analysis**

I. Introduction

Resource (demand and supply) forecasting in spatio-temporal data is widely studied in several fields including transportation [24], [30], construction [8], [10] and communication [41]. Traditional forecasting methods model (temporal) dependency of resource time-series values over time [28]. However, resource time-series values are typically recorded at various locations and exhibit dependencies across, and based on, locations. This requires additionally modeling relational and spatial dependencies. Moreover, contextual (environmental) factors (e.g., weather) also impact resource values and should be considered. However, existing methods that model context assume it has a fixed impact on resource values [14]. In practice, contexts have a time-evolving (dynamic) impact, which requires additionally modeling dynamic contextual dependency.

To address the aforementioned challenges, we propose a novel graph model, Context Integrated Graph Neural Network (**CIGNN**), which learns and incorporates temporal, relational, spatial, and dynamic contextual dependencies for resource forecasting. CIGNN considers a resource network of values recorded over time at locations, and models it as a graph, where nodes represent locations and the corresponding resource value time-series. CIGNN additionally uses separate graphs to represent each context type, where nodes represent context-recording locations and the associated contextual time-series.

CIGNN jointly predicts future values of resource and contextual time-series simultaneously, and leverages a novel fusion mechanism to incorporate contextual impact on resource values. To the best of our knowledge, this is the first work that integrates multiple sources of dynamic contextual information for spatio-temporal time-series prediction. We evaluate our model on two tasks: forecasting demand in a mobile call network and forecasting supply in a bike-sharing system on the *CallMi* and *BikeBay* datasets, respectively. The main contributions of this work can be summarized as follows:

- **Modeling Temporal, Relational, Spatial, and Dynamic Contextual Dependencies**: CIGNN performs resource forecasting by leveraging temporal, relational, spatial, and *dynamic* contextual dependencies. Existing methods do not capture dynamic context. Table I shows a qualitative comparison of CIGNN against state-of-the-art methods.
- **Unified Multi-source Context Learning**: In contrast to existing work that manually conducts feature engineering [49], treats context as fixed features [14], or ignores contextual information [6], [25], CIGNN integrates multiple contextual features in a unified way.
- **Effectiveness**: CIGNN consistently outperforms previous methods in terms of mean absolute error (MAE) and root mean square error (RMSE) on both datasets. CIGNN achieves average improvement of 5.7% (MAE) and 9.4% (RMSE) on *CallMi*; and 4.4% (MAE) and 2.3% (RMSE) on *BikeBay*.

II. Related Work

Classic time-series prediction methods (e.g., ARIMA and exponential smoothing) do not capture relational or spatial dependencies between nodes, cannot handle time-series with irregularities, and perform poorly on long-term forecasting [15], [28]. In contrast, deep learning-based approaches integrate complex non-linear dependency [2], [4], [31]. For example, previous spatio-temporal methods capture spatial dependency using convolutional neural networks (CNNs) and graph neural

TABLE I
QUALITATIVE COMPARISON: CIGNN IS THE ONLY MODEL THAT INCORPORATES TEMPORAL, RELATIONAL, SPATIAL, AND CONTEXTUAL DEPENDENCIES.

Modeled Dependency	ARIMA	VAR	LSTM	STGCN [50]	DCRNN [25]	Graph WaveNet [47]	**CIGNN**
TEMPORAL	✓	✓	✓	✓	✓	✓	✓
SPATIAL				✓	✓	✓	✓
RELATIONAL				✓	✓	✓	✓
DYNAMIC CONTEXTUAL							✓

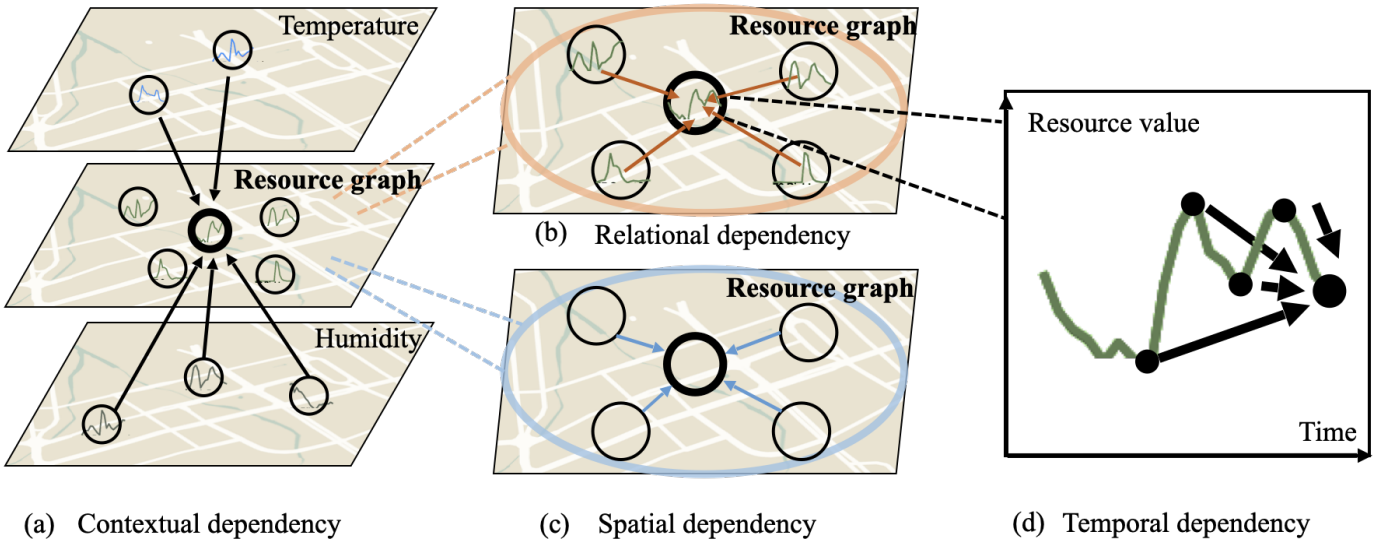

Fig. 1. A simplified illustration of spatial, temporal, relational, and contextual dependencies using only one node. (a) Contextual dependency: contextual (temperature and humidity) impact on resource graph. (b) Relational dependency: correlation between nodes in a graph. (c) Spatial dependency: spatial accessibility between nodes in a graph. (d) Temporal dependency: historical influence on future values within a single time-series.

networks (GNNs) [26], [29], [45]. Moreover, many recurrent neural network (RNN) variants have been used to capture temporal dependency [37], [44].

GNNs showed unprecedented advantages in time-series forecasting in several fields [17], [19], [33] such as traffic forecasting [6], [35], [42], [46], [51], crowd flow prediction [36], bike demand prediction [27], and weather prediction [43]. For example, GraphSAGE [16] and FastGCN [5] sample and aggregate neighborhood information for graph classification. Other work used GNNs for crime and traffic forecasting [40].

CNNs have also been used for time-series forecasting. For example, Li et al. used diffusion convolution to predict speed based on proximity information [25]. Zhang et al.proposed a Kernel-Weighted Graph Convolutional Network (KW-GCN) for traffic forecasting [52]. Other applications include video frame analysis [18], [22], which leverage the graph topology to model the human motion and object interactions. However, these methods have limited predictive accuracy because they do not utilize contextual information.

Contextual (or environmental) features significantly impact resource dynamics. For instance, ride-hailing demand is sensitive to precipitation. It is therefore imperative to integrate contextual factors into forecasting models. Related work on leveraging contextual features include TensorCast [9] which forecasts author collaboration by incorporating features in tensors. Other work utilized economic features to forecast housing price [13]. However, existing approaches assume *static contexts* [14], [32] or incur feature selection cost [49]. In contrast, CIGNN integrates dynamic contextual impact, simultaneously handles multiple types of contextual information, without incurring feature selection cost.

III. CONTEXT INTEGRATED GRAPH NEURAL NETWORK

We propose a novel *Context Integrated Graph Neural Network* (CIGNN) model, which leverages temporal, relational, spatial, and contextual dependencies for resource forecasting. CIGNN represents the resource network as a graph, where nodes represent locations and associated resource time-series. CIGNN also represents each context type as a separate graph, where nodes represent locations and associated contextual time-series. We introduce CIGNN using a bike-sharing system as a running example. Given historical bike supply records at various stations, CIGNN predicts future supply by considering the following dependencies (also illustrated in Fig. 1):

Definition 1 (Temporal dependency): Given time-series $\mathbf{x} = [x_1 x_2 \cdots x_T]$, temporal dependency is modeled as a function that maps values prior to x_t (e.g., previous w values) to x_t:

$$[x_{t-w} \cdots x_{t-2} x_{t-1}] \rightarrow x_t \tag{1}$$

TABLE II
SUMMARY OF NOTATION

Symbol	Description
E_r	edges with respects to relational dependency
E_s	edges with respects to spatial dependency
E_{sc}	edges with respects to contextual dependency
$\lvert\cdot\rvert$	the cardinality operator
$\mathcal{G}, M = \lvert\mathcal{G}\rvert$	a set of graphs and the number of graphs
T, w, h	length of time-series, window and horizon
G_i	the ith graph, where $i \in \{1, \cdots, M\}$
$V_i, E_i, \mathbf{A}_i$	nodes, edges and matrix in ith graph
$N_i = \lvert V_i\rvert$, d_i	number of nodes and features in ith graph
$\mathcal{X}_i$	the graph signal of the ith graph
$\mathcal{Y}$	a graph signal
$\mathbf{A}, \mathbf{D}$	Adjacency matrix and degree matrix
$\mathbf{L}, \mathbf{I}$	Laplacian matrix and the identity matrix
$\mathbf{r}^t, \mathbf{u}^t$	reset gate and update gate at timestamp t
$\mathbf{X}^t, \mathbf{H}^t$	input and hidden state at timestamp t
$\mathbf{FC}_*(\mathbf{X})$,	$\mathbf{W}_*^\intercal \mathbf{X} + \mathbf{b}_*$, a fully connected layer
$\mathbf{\Theta}_{*G}$	the graph convolution layer (on graph G)
$\mathbf{\Phi}$	our proposed fusion layer
$\theta, \mathbf{W}, \mathbf{b}, \mathbf{z}$	trainable parameters
$\sigma, tanh$	the sigmoid and tanh activation functions
$\oplus, \odot$	concatenation and Hadamard product operator

Modeling temporal dependency allows forecasting future bike supply based on previous values.

Next, we define relational and spatial dependencies to model mutual influence across supply at various locations:

Definition 2 (Relational dependency): Given a graph $G = (V, E)$, where V denotes the node set and E the edge set. Each node represents a station with an associated resource (in this case, the supply) time-series. Then, we define edges to represent relational dependency:

$$E = \{(i,j) \mid \forall (i,j) \in |V| \times |V|, \textit{s.t. } \mathbb{K}(\mathbf{x}_i, \mathbf{x}_j) \geq \lambda_r\} \quad (2)$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ denote the supply time-series in node i and j, respectively. $\mathbb{K}$ denotes a correlation metric and λ_r a threshold value to filter out small correlation values.

Edges with a weight $\mathbb{K}(\mathbf{x}_i, \mathbf{x}_j)$ imply relational dependency while edges are removed if their weights are smaller than a threshold λ_r. The insight behind relational dependency is to model the mutual dependencies between similarly behaving time-series.

On the other hand, the spatial dependency is constructed differently, based on the first law of geography [39], i.e., *Near things are more related than distant things.*

Definition 3 (Spatial dependency): Given a graph $G = (V, E, \mathbf{A})$ where V denotes the node set and E the edge set. Each node represents a biking station. The adjacency matrix $\mathbf{A}$ encodes the distance information between nodes. We define the edges to represent spatial dependency:

$$E = \{(i,j) \mid \forall (i,j) \in |V| \times |V|, \textit{s.t. } \mathbf{A}_{ij} \geq \lambda_s\} \quad (3)$$

$\mathbf{A}_{ij}$ is derived from distance information such that a short distance derives a large weight value. Edges with a large weight indicate spatial dependency when the weights are greater than a threshold λ_s.

Definition 4 (Contextual dependency): Let $G_s = (V_s, E_s)$ denote a resource graph with a set of nodes V_s representing the bike stations that connected by edges in E_s. In addition, let $G_c = (V_c, E_c)$ be a context graph with a set of nodes V_c representing the locations where contextual features are recorded (e.g., weather stations recording humidity), and connected by edges E_c. Note that E_s and E_c are edges derived either from Definition 2 or Definition 3. We denote E_{sc} as edges connecting nodes in V_s with nodes in V_c.

$$E_{sc} = \{(i,j) \mid \forall (i,j) \in |V_s| \times |V_c|\} \quad (4)$$

To account for more than one contextual type (e.g., if there are n contextual types), the definition can be extended to include all contextual types: $V_c = V_{c1} \cup \ldots \cup V_{cn}$, $E_c = E_{c1} \cup \ldots \cup E_{cn}$, for contextual types $c1, \ldots, cn$.

In a bike-sharing system, the bike supply is sensitive to weather conditions such as temperature and precipitation, which are recorded in weather stations. In this case, we build a contextual graph for temperature information and one for precipitation.

A. Problem Formulation

We formulate the resource forecasting in spatio-temporal data as a time-series prediction task where time-series are recorded in various locations. The task aims to learn a function f that predicts future resource values given spatio-temporal information, which is organized in a form of graph. Each node in a graph represents time-series, and edges connecting nodes represent a distance or a correlation computed as a function of spatial or relational dependencies. The resource network and the $M-1$ contextual networks are represented as a set of M graphs: $\mathcal{G} = \{G_1, G_2, \ldots, G_M\}$. Each graph either belongs to the resource (demand/supply) or a contextual type. For instance, to forecast the bike supply, we derive a bike supply graph, a contextual graph regarding the temperature and a contextual graph regarding the humidity. The ith graph is denoted as $G_i = (V_i, E_i, \mathbf{A}_i)$, where V_i represent the node set and E_i the edge set in G_i. Nodes represent locations with associated time-series and edges represent the dependency between nodes. $\mathbf{A}_i$ is the adjacency matrix that encodes a correlation computed as a function of spatial or relational dependencies.

For convenience, we refer to the group of time-series in a graph as a *graph signal* [12], which contains all time-series from the same (graph) type. Thus for graph G_i, the graph signal is denoted as $\mathcal{X}_i \in \mathbb{R}^{T \times N_i \times d_i}$, where T denotes the time span of interest, $N_i = |V_i|$ is the number of nodes in the ith graph, d_i is the corresponding number of node features. We use a subscript to denote which graph is referred, and a superscript to denote the time index. For example, $\mathcal{X}_i^t \in \mathbb{R}^{N_i \times d_i}$ is the graph signal of G_i at t.

Given graphs and graph signals, a multi-step ahead time-series prediction task regarding w past observations and a

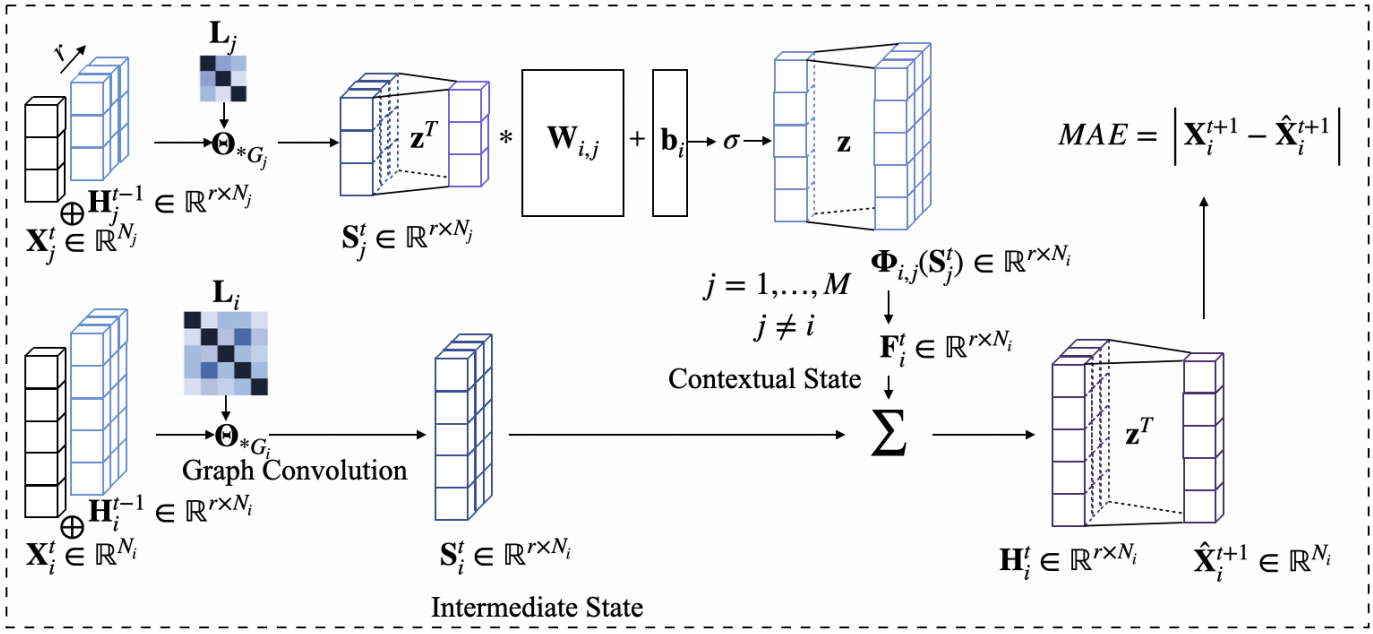

Fig. 2. An overview of our CIGNN unit. Due to space limit, only two graphs are displayed in the figure and number of node features are assumed as one, $d_i = d_j = 1$. The ith graph is regarded as the resource graph and the jth graph is regarded as a contextual graph. The graph convolution is denoted by $\mathbf{\Theta}$ and is used to capture the relational and spatial dependencies. The intermediate state $\mathbf{S}$ incorporates the temporal dependency. The contextual state $\mathbf{F}$ is combined for the contextual dependency modeling. In practice, CIGNN learns $\mathbf{\Phi}_{i,j}$ for $\{(i,j)|i,j \in 1,2,...,M, i \neq j\}$.

horizon h is formulated as:

$$[\mathcal{X}_i^{t-w+1}\mathcal{X}_i^{t-w+2}\ldots\mathcal{X}_i^{t};G_i] \xrightarrow{f(\cdot)} [\mathcal{X}_i^{t+1}\mathcal{X}_i^{t+2}\ldots\mathcal{X}_i^{t+h}] \quad i \in [1,\ldots,M]$$

Our model architecture is depicted in Fig. 2. We introduce our components for modeling the relational, spatial, temporal and contextual dependency in the following sections.

B. Relational Dependency Modeling using Graph Convolution

The relational dependency is the implicit connections between stations. We leverage the graph convolution to model it. Given a graph $G(V,E,\mathbf{A})$ (where $\mathbf{A} \in \mathbb{R}^{N\times N}$ is constructed in a way such that it follows the Definition 2) and its graph signal $\mathcal{Y} \in \mathbb{R}^{N\times d}$ with $N = |V|$ and d the number of features, we define a graph convolution layer $\mathbf{\Theta}$ regarding G as:

$$\begin{aligned}\mathbf{\Theta}_{*G}\mathcal{Y} &= \mathbf{\Theta}(\mathbf{L})\mathcal{Y} \\ &= \mathbf{\Theta}(\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\mathsf{T})\mathcal{Y} \\ &= \mathbf{Q}\mathbf{\Theta}(\mathbf{\Lambda})\mathbf{Q}^\mathsf{T}\mathcal{Y}\end{aligned} \tag{5}$$

where $\mathbf{L} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{D}-\mathbf{A})\mathbf{D}^{-\frac{1}{2}}$ is the normalized Laplacian matrix of adjacency matrix $\mathbf{A}$, with $\mathbf{D}$ its diagonal degree matrix. $\mathbf{L} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\mathsf{T}$ is the Eigen-decomposition of $\mathbf{L}$, where $\mathbf{Q}$ is composed of eigenvectors. $\mathbf{\Lambda}$ is a diagonal matrix where each element is an eigenvalue of the corresponding eigenvector.

Chebyshev Approximation is applied to strengthen locality and boost computational efficiency. The intuition behind leveraging graph convolution is that it is efficient on aggregating information from neighboring nodes. Further, we enhance the connection from local neighboring nodes by extending the graph kernel $\mathbf{\Theta}(\mathbf{\Lambda})$ to a series of bases as:

$$\mathbf{\Theta}(\mathbf{\Lambda}) = \sum_{k=0}^{K-1}\theta_k\mathbf{\Lambda}^k \tag{6}$$

where $\boldsymbol{\theta} = [\theta_1\theta_2\ldots\theta_k]^\mathsf{T} \in \mathbb{R}^K$ is a trainable coefficient vector and K is a value set to regulate the locality radius of graph convolution. We set a small number for K to reinforce the local impacts. The truncated Chebyshev polynomial expansion [11], [38] is then adapted with Eq. 6:

$$\mathbf{\Theta}_{*G}\mathcal{Y} = \mathbf{\Theta}(\mathbf{L})\mathcal{Y} \approx \sum_{k=0}^{K-1}\theta_k T_k(\tilde{\mathbf{L}})\mathcal{Y} \tag{7}$$

$$\tilde{\mathbf{L}} = \frac{2\mathbf{L}}{\lambda_{max}} - \mathbf{I} = \mathbf{L} - \mathbf{I}, \text{assuming } \lambda_{max} = 2 \tag{8}$$

where $T_k(\tilde{\mathbf{L}})$ is the kth Chebyshev polynomial at the scaled Laplacian $\tilde{\mathbf{L}}$, and $\mathbf{I} \in \mathbb{R}^{N\times N}$ is the identity matrix. λ_{max} is the dominant eigenvalue of $\mathbf{L}$, which is assumed as 2 since parameters can adapt to the change in scale during training [21]. The Chebyshev polynomial reduces the time complexity from $\mathcal{O}(N^3)$ to $\mathcal{O}(K|\mathcal{E}|)$ and accelerates computations, where $\mathcal{E}$ is the number of non-zero edges. Recall that we formulate a resource graph and multiple contextual graphs (M graphs in total), the graph convolution is applied on each graph with its individual parameters. For the ith graph, the graph convolution is performed with the following formula:

$$\mathbf{\Theta}_{*G_i}(\mathcal{X}_i) = \sum_{k=0}^{K-1}\theta_{k,i}T_k(\tilde{\mathbf{L}})\mathcal{X}_i, \quad i = 1,2,\ldots,M \tag{9}$$

C. Spatial Dependency Modeling using Graph Convolution

Alternatively, the mutual impact between locations can be modeled with spatial dependency, defined in Definition 3. In this situation, the adjacency matrix in the graph describes the geographic proximity between nodes. The graph convolution is then conducted with these differently constructed graphs.

D. Temporal Dependency Modeling using Gated Recurrent Units

Inspired by the recent success of RNN structure in time-series prediction and its power of temporal dependency capturing, we leverage a modified Gated Recurrent Units (GRU) [7] variant of the RNN methods in our approach. The temporal dependency modeling component in CIGNN is composed of units that take current observations at time t from each graph $\mathcal{X}_i^t \in \mathbb{R}^{N_i \times d_i}$ and previous hidden states $\mathbf{H}_i^{t-1} \in \mathbb{R}^{r \times N_i \times d_i}$ as input (where $i \in [1 \dots M]$), then yields current hidden states at time t as $\mathbf{H}_i^t \in \mathbb{R}^{r \times N_i \times d_i}$ as output, (where r is the number of neurons). Our GRU-like structure is formulated as follows:

$$\mathbf{r}_i^t = \sigma(\mathbf{FC}_r(\mathbf{\Theta}_{r*G_i}[\mathcal{X}_i^t \oplus \mathbf{H}_i^{t-1}])) \tag{10}$$

$$\mathbf{u}_i^t = \sigma(\mathbf{FC}_u(\mathbf{\Theta}_{u*G_i}[\mathcal{X}_i^t \oplus \mathbf{H}_i^{t-1}])) \tag{11}$$

$$\mathbf{C}_i^t = \tanh(\mathbf{FC}_C(\mathbf{\Theta}_{C*G_i}[\mathcal{X}_i^t \oplus (\mathbf{r}_i^t \odot \mathbf{H}_i^{t-1})])) \tag{12}$$

$$\mathbf{S}_i^t = \mathbf{u}_i^t \odot \mathbf{H}_i^{t-1} + (1 - \mathbf{u}_i^t) \odot \mathbf{C}_i^t \tag{13}$$

where $\mathbf{r}$ and $\mathbf{u}$ denote the reset gate and the update gate, respectively. σ and $\tanh$ denotes the sigmoid and hyperbolic tangent activation functions, respectively. $\mathbf{FC}_*(\mathbf{X}) = \mathbf{W}_*^\mathsf{T}\mathbf{X} + \mathbf{b}_*$ is short for a fully connected layer. The operators $\oplus$ and $\odot$ denote concatenation and Hadamard product, respectively. $\mathbf{\Theta}$ is the graph convolution layer as introduced in Sec. III-B, which captures either relational dependency or spatial dependency, based on the construction of the adjacency matrix. The intermediate hidden state $\mathbf{S}_i$ incorporates both temporal, relational or spatial dependency in the ith graph.

E. Contextual Dependency Modeling with Novel Fusion Layers

With intermediate hidden states from the previous section, we further propose a novel fusion layer $\mathbf{\Phi}$ to capture the contextual dependency across graphs as defined in Definition 4. Our fusion mechanism, in contrast to existing work, does not require manual feature engineering:

$$\mathbf{\Phi}_{i,j}(\mathbf{S}_j^t) = \sigma(\mathbf{z}[\mathbf{W}_{i,j}^\mathsf{T}\mathbf{z}^\mathsf{T}\mathbf{S}_j^t + \mathbf{b}_{i,j}]) \tag{14}$$

$$\mathbf{F}_i^t = \sum_{j=1,\, j\neq i}^{M} \mathbf{\Phi}_{i,j}(\mathbf{S}_j^t) \tag{15}$$

where the subscript denotes different sets of parameters. The trainable parameters $\mathbf{W}_{i,j} \in \mathbb{R}^{N_j \times d_j \times d_i \times N_i}, \mathbf{b}_{i,j} \in \mathbb{R}^{N_i \times d_i}$ are regarded as learning relations between time-series across graphs, i.e., the contextual dependency. Note that $\mathbf{W}$ has a similar form as E_{sc} defined in Eq. 4 when we consider the resource graph (e.g., the bike supply network) and its contextual graphs (e.g., the temperature graph and the humidity graph). The parameter $\mathbf{z} \in \mathbb{R}^r$ is a mapping trainable vector that can be considered as aggregating neurons.

Finally, the hidden states $\mathbf{H}_i^t$ combine the intermediate hidden state $\mathbf{S}_i^t$ and the fused hidden state $\mathbf{F}_i^t$:

$$\mathbf{H}_i^t = \mathbf{S}_i^t + \mathbf{F}_i^t \tag{16}$$

F. Prediction and Objective

With the hidden state $\mathbf{H}_i^t$, the forecasting is conducted as:

$$\hat{\mathcal{X}}_i^{t+1} = \mathbf{z}^\mathsf{T}\mathbf{H}_i^t \tag{17}$$

while forecasts on further values (e.g., $\hat{\mathcal{X}}_i^{t+2}, \hat{\mathcal{X}}_i^{t+3}, etc.$) are calculated recursively. An objective function is designed to be minimized during the training stage:

$$\underset{\theta,\mathbf{W},\mathbf{b},\mathbf{z}}{\operatorname{argmin}} \sum_{i=1}^{M}\sum_{j=1}^{h} \left| \hat{\mathcal{X}}_i^{t+j} - \mathcal{X}_i^{t+j} \right|, \tag{18}$$

which minimizes the mean absolute error between the prediction and the actual values in all training samples across graphs and time horizons.

IV. Experiments

In this section, we introduce our experimental setup which includes how graphs are constructed and the hyperparameters. To verify the effectiveness of CIGNN, two real-world datasets are selected and previous state-of-the-art methods are used for comparison.

A. Graph Construction

The graph construction decides what information and dependency is modeled. We introduce two existing ways to construct a graph. One is based on geographical information as in Definition 3 to model the spatial dependency, the other is based on the implicit correlations between time-series as in Definition 2. Unless otherwise stated, we denote the adjacency matrix as $\mathbf{A} \in \mathbb{R}^{N \times N}$, and $\mathbf{A}_{ij}$ is an element.

a) Distance-based Gaussian Kernel Matrix: Most existing work [25] derives an adjacency matrix with a truncated Gaussian kernel, such that the element $\mathbf{A}_{ij}$ has higher values if node i and node j are close in terms of their locations:

$$\mathbf{A}_{ij} = \begin{cases} exp(-\frac{d(i,j)^2}{\sigma^2}), & \text{if } d(i,j) \leq \kappa \\ 0, & \text{otherwise} \end{cases} \tag{19}$$

where $d(i,j)$ denotes the geographical distance between nodes i and j. σ denotes the standard deviation of distances and κ is a threshold value (set as 0.1) to control the matrix sparsity. Note that $\lambda_s = exp(-\frac{\kappa^2}{\sigma^2})$ as in Definition 3.

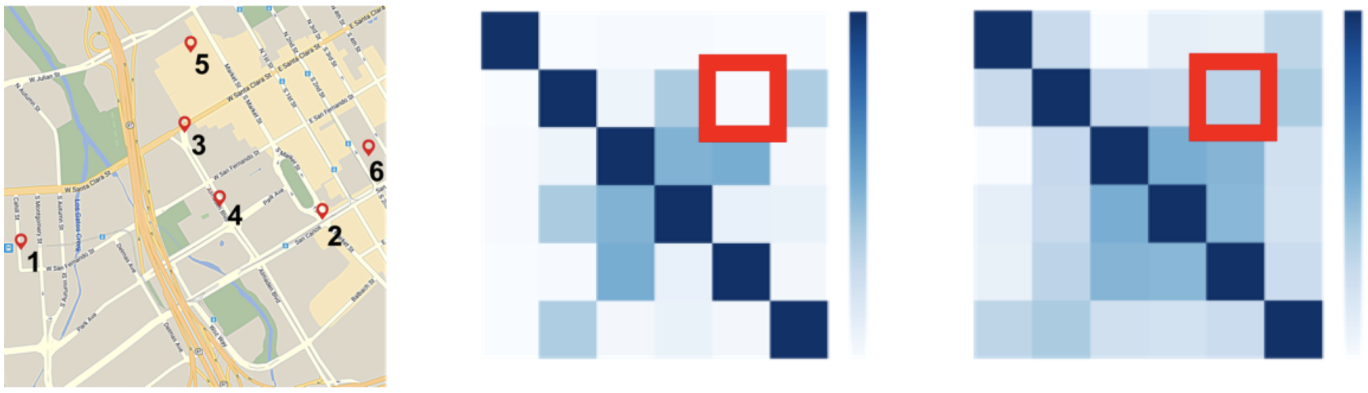

Fig. 3. A comparison between Gaussian kernel matrix and Relational matrix. Dense color indicates higher correlations. (a). Some bike docks in San Jose. (b). The Gaussian kernel matrix. (c). The relational correlation matrix. The value between station 2 and 5 (marked by the red squares) is higher.

TABLE III
DATASET DESCRIPTIONS: A GRAPH IS CONSTRUCTED FOR THE DEMAND OR SUPPLY, AND FOR EACH TYPE OF CONTEXTUAL FEATURES.

Dataset	**CallMi**	**BikeBay**
Data meaning	Mobile call records	Bike supply amount
Data location	Milan city	The Bay Area
Number of nodes	162	70
Time span	Nov. 2013-Dec. 2013	Aug. 2013- Aug. 2015
Time interval	1 hour	2 hour
Contextual types (number of nodes)	temperature(5) humidity (4)	temperature (3), humidity(3),dew(3) sea level(3) and wind speed(3)

b) Relational Matrix based on Correlation Coefficients: We observe in empirical studies that the Gaussian kernel matrix can fail to capture the hidden relational correlations between distant nodes. For example, in Fig. 3a, station 2 and station 5 are far from each other. The Distance-based Gaussian Kernel matrix (Fig. 3b) suggests they have low correlations. However, they are highly correlated (Fig. 3c) in reality since people frequently commute between them along the straight road. To reflect this fact, we utilize the Detrended Cross-Correlation Analysis coefficient (DCCA coefficient) [23], [48] to construct the *relational correlation matrix*.

The DCCA coefficient is a correlation metric on series data, which combines detrended cross-correlation analysis (DCCA) and detrended fluctuation analysis (DFA). Given two time-series $\mathbf{x}, \mathbf{y} \in \mathbb{R}^T$ of length T and sliding windows of length l, the DCCA coefficient is defined as:

$$\rho_{DCCA}(\mathbf{x}, \mathbf{y}, l) = \frac{F^2_{DCCA}(\mathbf{x}, \mathbf{y}, l)}{F_{DFA}(\mathbf{x}, l) F_{DFA}(\mathbf{y}, l)} \tag{20}$$

where the numerator and the denominator are the average covariance and variance of the $T - l + 1$ windows (partial sums):

$$F^2_{DCCA}(\mathbf{x}, \mathbf{y}, l) = \frac{\sum_{s=1}^{T-l+1} f^2_{DCCA}(\mathbf{x}, \mathbf{y}, s)}{T - l + 1} \tag{21}$$

$$F^2_{DFA}(\mathbf{x}, l) = \frac{\sum_{s=1}^{T-l+1} f^2_{DFA}(\mathbf{x}, s)}{T - l + 1} \tag{22}$$

The partial sums are calculated with sliding windows across $\mathbf{x}$ and $\mathbf{y}$. For each window with the starting index s:

$$f^2_{DCCA}(\mathbf{x}, \mathbf{y}, s) = \frac{\sum_{t=s}^{s+l-1} (\mathbf{x}^t - \bar{\mathbf{x}}_s)(\mathbf{y}^t - \bar{\mathbf{y}}_s)}{l} \tag{23}$$

$$f^2_{DFA}(\mathbf{x}, s) = \frac{\sum_{t=s}^{s+l-1} (\mathbf{x}^t - \bar{\mathbf{x}}_s)^2}{l} \tag{24}$$

where $\bar{\mathbf{x}}_s = \frac{1}{l}(x_s + x_{s+1} + \ldots + x_{s+l-1})$ is the average value in the window started at s. The matrix is constructed with the pairwise correlations:

$$\mathbf{A}_{ij} = \begin{cases} \rho_{DCCA}(\mathbf{x}, \mathbf{y}, l), & \text{if } \rho_{DCCA}(\mathbf{x}, \mathbf{y}, l) \geq \lambda_r = 0 \\ 0, & \text{otherwise} \end{cases}$$

B. Experimental Setup

We conduct experiments on two public real-world datasets (see Table III for a summary) to show the performance of CIGNN. Both datasets contain dynamic contextual features.

- **Mobile Call Demand in Milan *(CallMi)*** [3]: *CallMi* contains call demand data in Milan from Nov. 2013 to

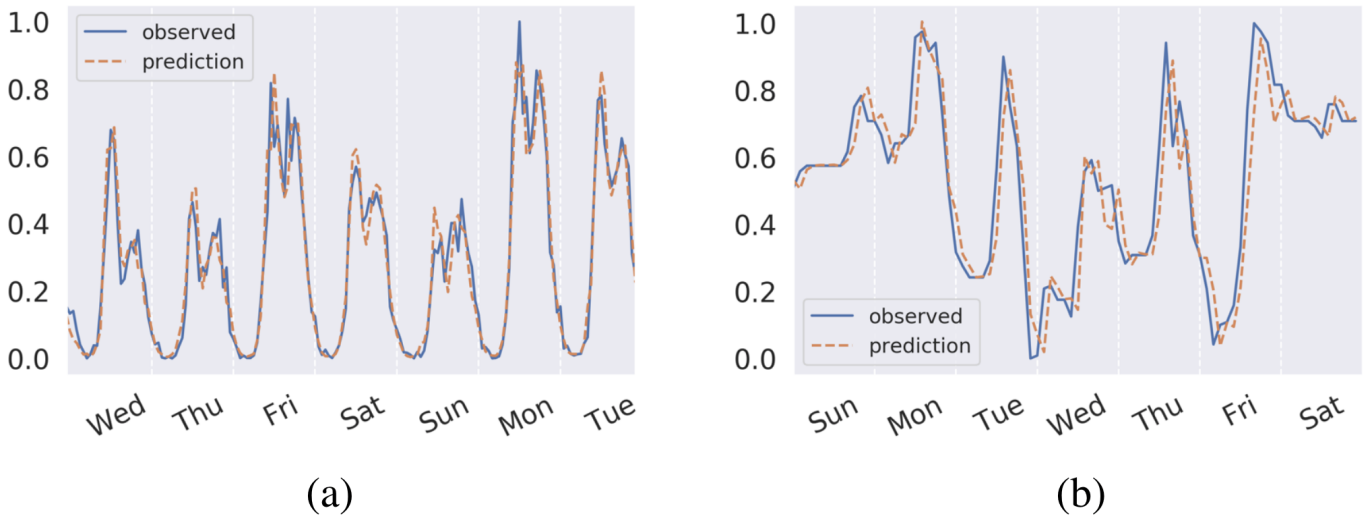

Fig. 4. A comparison of time-series pattern. (a) The call demand during Dec. 25-31. (b) The bike supply in San Francisco during Aug. 25-31. Values are normalized for display. Note that *CallMi* shows more periodicity.

TABLE IV
A COMPARISON USING 6 OBSERVED DATA POINTS TO PREDICT 3 STEPS AHEAD. MAE AND RMSE OF THREE HORIZONS, THEIR AVERAGE, AND AVERAGE ERROR REDUCTION OVER VAR AS A BASELINE FOR *CallMi* AND *BikeBay*.

					CallMi				
Horizon	Metrics	HA	ARIMA	VAR	LSTM	STGCN	DCRNN	Graph WaveNet	**CIGNN**
1	MAE	17.15	14.42	18.54	13.51	11.35	10.41	9.48	**8.89**
	RMSE	38.80	24.26	31.30	25.04	20.46	19.44	18.18	**16.82**
2	MAE	–	26.78	27.01	17.05	20.48	16.59	12.72	**11.72**
	RMSE	–	44.27	47.14	30.10	35.63	33.59	26.23	**22.84**
3	MAE	–	38.12	34.49	19.02	33.14	22.60	15.38	**14.86**
	RMSE	–	61.43	60.13	35.04	40.01	55.03	32.09	**29.59**
					BikeBay				
Horizon	Metrics	HA	ARIMA	VAR	LSTM	STGCN	DCRNN	Graph WaveNet	**CIGNN**
1	MAE	22.70	7.62	8.04	19.91	6.80	6.55	7.00	**6.37**
	RMSE	28.92	13.35	18.72	25.34	11.98	12.37	13.33	**11.75**
2	MAE	–	11.70	12.06	20.83	10.76	10.13	11.01	**9.68**
	RMSE	–	18.62	29.42	26.67	16.98	17.65	19.56	**16.60**
3	MAE	–	14.12	14.34	21.29	13.41	12.56	13.69	**11.90**
	RMSE	–	21.19	35.05	27.73	19.71	20.52	22.25	**19.18**

Dec. 2013. The dataset contains temperature and humidity as contextual features. The city is partitioned into grids in the raw dataset, however, some grids have very few records. Therefore, we cluster grids into 162 mobile call nodes/instances. There are 5 temperature nodes and 4 humidity nodes. Each of contextual nodes corresponds to a weather station. The time interval is an hour.

- **Bike-sharing Supply in the Bay Area *(BikeBay)*** [1]: *BikeBay* contains the bike supply data in 70 dock stations in the Bay Area. The dataset was recorded from Aug. 2013 to Aug. 2015. It contains weather conditions as contextual data. There are 3 nodes for temperature, humidity, dew, sea level, and wind speed, respectively. The time interval is two hours.

We selected these two datasets for the reason they exhibit different time-series patterns. The following hyperparameters are being used: 0.01 (learning rate), 32 (number of neurons), 0.1 (learning rate decay ratio for every 10 epochs). We train for a maximum of 100 epochs with the Adam optimizer [20] and adapt an early stop strategy if the validation loss does not decrease for 10 consecutive epochs. All experiments are implemented using Python Tensorflow (v1.14) and run on Ubuntu 16.04 OS with 8 CPU cores and a memory of 32G.

a) Diversity of Datasets regarding Periodicity.: The two datasets demonstrate different pattern characteristics, as shown in Fig. 4. The call demand time-series exhibit periodicity (Fig. 4a), while the bike supply time-series show more irregularities (Fig. 4b). We choose these two datasets to demonstrate

CIGNN's capability to predict accurately on time-series with or without periodicity.

The time-series data are split chronologically into training, validation and testing sets in a ratio of 70%:10%:20%. Values are normalized using Z-score normalization before training. Predicted values are inversely transformed to evaluate error. The number of lags w and horizons h are set as 6 and 3, respectively. We conducted experiments on both the Gaussian kernel matrix and the DCCA coefficient relational matrix. When using the relational matrix, the window length l in Eq. 23 is set as 4. A previous study [34] shows that a small step of graph convolution is more effective, thus graph convolution step in Eq. 9 is set as $K = 1$ to strengthen the local impacts. Mean Absolute Error (MAE) is used as a loss measure to update parameters for all horizons in the training set. In evaluation, both MAE and Root Mean Square Error(RMSE) are calculated and compared:

$$MAE = \frac{1}{|\mathbf{\Omega}|} \sum_{t \in \mathbf{\Omega}} \sum_{i=1}^{M} |\mathcal{X}_i^t - \hat{\mathcal{X}}_i^t|$$

$$RMSE = \sqrt{\frac{1}{|\mathbf{\Omega}|} \sum_{t \in \mathbf{\Omega}} \sum_{i=1}^{M} (\mathcal{X}_i^t - \hat{\mathcal{X}}_i^t)^2}$$

where $\mathbf{\Omega}$ denotes the timestamps of measured samples (e.g., Ω denotes the set of training samples when in the training stage).

C. Baseline Methods

We compare CIGNN to the following state-of-the-art methods. We do not compare to [49] since it is only applicable to grid-formatted data.

1) **HA** Historical Average. The prediction for a given timestamp is the average of previous values at that same timestamp (and day) over the past four weeks.
2) **ARIMA** Auto-Regressive Integrated Moving Average. The ARIMA learns from each time-series and is used to predict each time-series, independently. The ARIMA order is $(3, 0, 1)$, as in [25].
3) **VAR** Vector Auto-Regressive is a multi-variate model that generalizes ARIMA to have multiple evolving variables.
4) **MM-LSTM** Multi-step multi-variate LSTM. The number of neurons is set as 64.
5) **DCRNN** [25]: Diffusion Convolution Recurrent Neural Network models spatial correlations with a diffusion process. For the temporal correlations, DCRNN utilizes an encoder-decoder structure built by GRU units.
6) **STGCN** [50]: Spatio-Temporal Graph Convolutional Network is a graph-based model that learns both the spatial and temporal dependencies with gating mechanism to make predictions.
7) **Graph WaveNet** [47] Graph WaveNet aims to capture the long-term trend to make predictions. It leverages a self-adaptive adjacency matrix design to exploit spatial dependencies.

TABLE V
A COMPARISON OF DEEP LEARNING METHODS USING 24 OBSERVED DATA POINTS TO PREDICT 6 STEPS AHEAD ON *BikeBay*. G-WAVENET IS SHORT FOR WAVENET.

		BikeBay			
H	Metrics	STGCN	DCRNN	G-WaveNet	CIGNN
1	MAE	7.50	6.44	19.24	**6.38**
	RMSE	12.37	11.99	26.75	**11.80**
2	MAE	10.71	9.72	19.77	**9.60**
	RMSE	16.63	16.72	27.38	**16.47**
3	MAE	13.11	11.91	20.32	**11.75**
	RMSE	19.33	19.33	28.02	**19.01**
4	MAE	16.62	13.67	20.68	**13.47**
	RMSE	22.77	21.22	28.36	**20.85**
5	MAE	16.44	15.06	20.94	**14.83**
	RMSE	23.29	22.64	28.57	**22.24**
6	MAE	17.56	16.12	21.12	**15.90**
	RMSE	24.83	23.67	28.68	**23.31**

D. Results

a) Comparisons for Multi-step Ahead Prediction: Following common practice [25], [47], [50], we use six past observations to predict values that are three steps ahead. Results are shown in Table IV. MAE and RMSE are evaluated for each horizon and on the average across horizons. Using dataset *BikeBay*, we further conduct more experiments with more data points (24) and a greater horizon (6) on the deep learning methods, as shown in Table V. We observe the following:

- Deep learning based methods outperform traditional methods (HA, ARIMA and VAR) due to the latter methods' limit of only modeling temporal dependency. Besides, HA can only predict one step ahead, and ARIMA fails to exploit the interactions across time-series.
- For deep learning methods, WaveNet outperforms DCRNN on CallMi but not on BikeBay, while CIGNN consistently outperforms STGCN, DCRNN and WaveNet. CIGNN also outperforms baselines when more observations and a larger horizon are considered.
- CIGNN outperforms previous best state-of-the-art models on each dataset (improves over WaveNet on CallMi by 5.7% MAE and 9.4% RMSE and over DCRNN on BikeBay by 4.4% MAE and 2.3% RMSE).

The result shows that our model is capable to capture the contextual dependency of the dynamic demand.

b) Effectiveness of Fusion: To assess the effectiveness of our proposed fusion mechanism, we compared our approach to a variant that removes the fusion layer. We ran experiments on the BikeBay dataset and compared the two model for both training and testing loss, as shown in Fig. 5. Although CIGNN is initialized with a higher loss, its loss decreases faster. This shows that the fusion mechanism is effective in utilizing contextual factors on forecasting. We analyze the effectiveness of the Gaussian Kernel matrix and Relational matrix, added in appendix due to the limit of space.

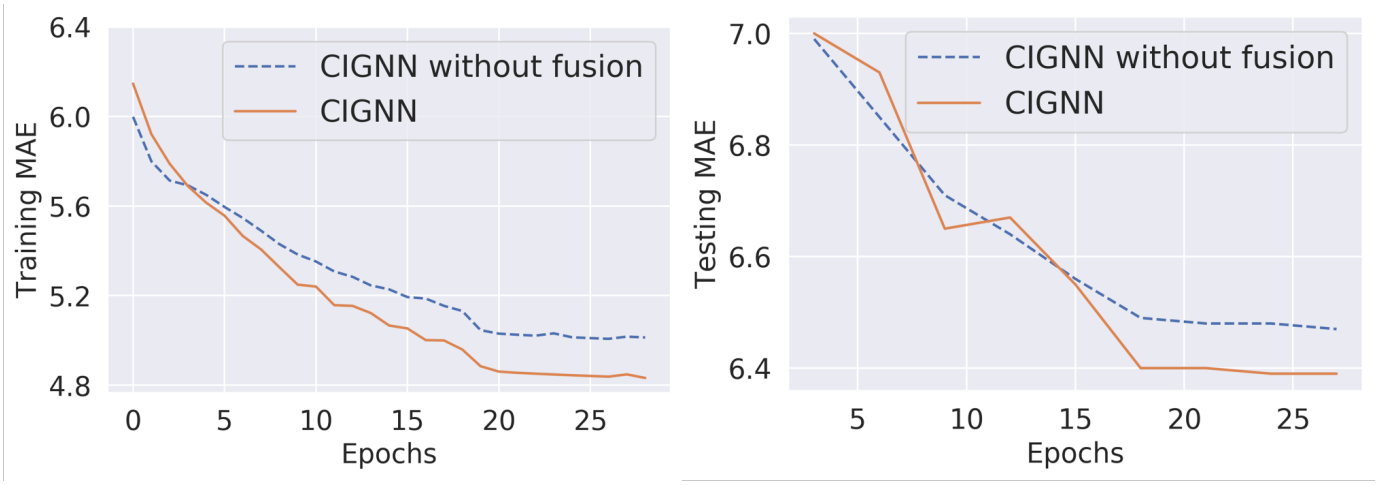

Fig. 5. An illustration of the effectiveness of the fusion mechanism. CIGNN with fusion outperforms the model without fusion (dashed), in both training and testing.

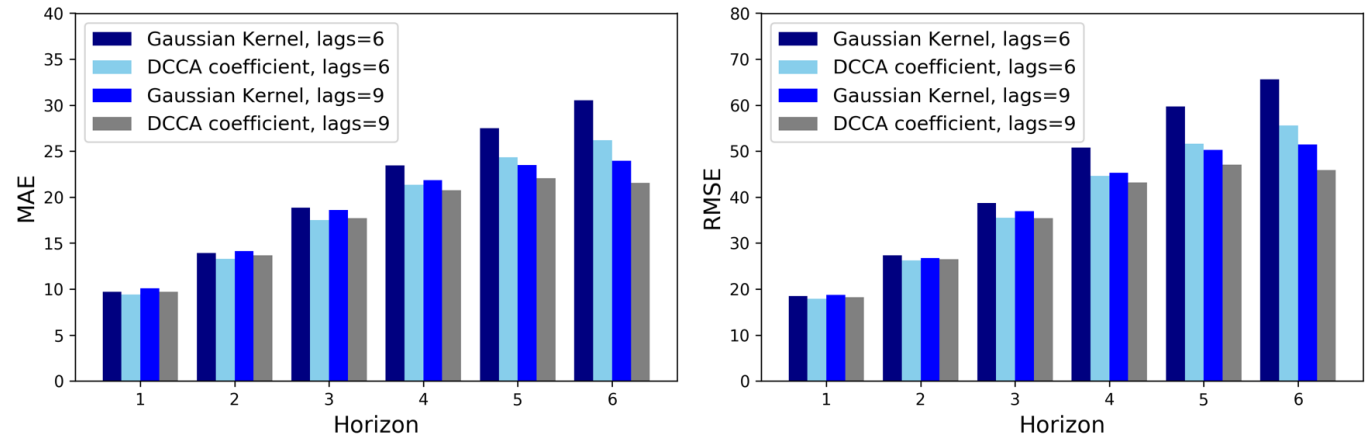

Fig. 6. MAE and RMSE comparison of models with adjacency matrix constructed by Gaussian Kernel and DCCA coefficient. DCCA coefficient-based model achieves better performance in both situations; when lag number is 6 or 9.

c) Adjacency Matrix Analysis.: To analyze and compare the effectiveness of the Gaussian Kernel matrix and the DCCA relational coefficients matrix, we ran experiments on the CallMi dataset using temporal lags 6 and 9. Fig. 6 shows the MAE and RMSE for a horizon of 6. The results show that: (1) Predictions based on the DCCA coefficient adjacency matrix are consistently more accurate than predictions based on the Gaussian kernel matrix. (2) The lag number has an impact on predicting values for a long horizon. Using 9 lags results in significantly better prediction results than using 6 lags. This demonstrates that CIGNN is better at learning long-term temporal dependencies.

V. Conclusion

To model non-linear temporal, relational, spatial, and contextual dependency in time-series predictions, we propose a novel Graph Neural Network approach for spatio-temporal data with dynamic contextual information. Our model employs a novel fusion mechanism to capture the dynamic contextual impact on demand.

The capability of processing multiple graphs at the same time empowers our model to be be applied in more general structured sequence forecasting scenarios, such as dynamic social networks relationship prediction, and evolving preference prediction in recommendation systems. To extend this work, we will theoretically analyze the interpretability of CIGNN model components (fusion layer weights distribution and DCCA relational coefficients matrix limits), apply CIGNN on other spatio-temporal data, explore more general graph signal representations, and consider more complex contextual structures for the fusion of static and dynamic contextual impact on resource fluctuations.

References

[1] Huthaifa I. Ashqar, Mohammed Elhenawy, and Hesham A. Rakha. Modeling bike counts in a bike-sharing system considering the effect of weather conditions. *Case Studies on Transport Policy*, 2019.

[2] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. Adaptive graph convolutional recurrent network for traffic forecasting. *arXiv preprint arXiv:2007.02842*, 2020.

[3] Gianni Barlacchi, Marco De Nadai, Roberto Larcher, Antonio Casella, Cristiana Chitic, Giovanni Torrisi, Fabrizio Antonelli, Alessandro Vespignani, Alex Pentland, and Bruno Lepri. A multi-source dataset of urban life in the city of milan and the province of trentino. *Scientific Data*, 2015.

[4] Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, et al. Spectral temporal graph neural network for multivariate time-series forecasting. *Advances in Neural Information Processing Systems*, 33, 2020.

[5] Jie Chen, Tengfei Ma, and Cao Xiao. Fastgcn: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247*, 2018.

[6] Weiqi Chen, Ling Chen, Yu Xie, Wei Cao, Yusong Gao, and Xiaojie Feng. Multi-range attentive bicomponent graph convolutional network for traffic forecasting. *arXiv preprint arXiv:1911.12093*, 2019.

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv:1412.3555*, 2014.

[8] Csaba Csiszár, Bálint Csonka, Dávid Földes, Ervin Wirth, and Tamás Lovas. Urban public charging station locating method for electric vehicles based on land use approach. *Journal of Transport Geography*, 2019.

[9] Miguel Ramos de Araujo, Pedro Manuel Pinto Ribeiro, and Christos Faloutsos. Tensorcast: Forecasting with context using coupled tensors (best paper award). *ICDM*, pages 71–80, 2017.

[10] Chirag Deb, Fan Zhang, Junjing Yang, Siew Eang Lee, and Kwok Wei Shah. A review on time series forecasting techniques for building energy consumption. *Renewable and Sustainable Energy Reviews*, 2017.

[11] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, 2016.

[12] Satoshi Furutani, Toshiki Shibahara, Mitsuaki Akiyama, Kunio Hato, and Masaki Aida. Graph signal processing for directed graphs based on the hermitian laplacian. In *ECML-PKDD*, 2019.

[13] Chuancai Ge, Yang Wang, Xike Xie, Hengchang Liu, and Zhengyang Zhou. An integrated model for urban subregion house price forecasting: A multi-source data perspective. In *ICDM*. IEEE, 2019.

[14] Xu Geng, Yaguang Li, Leye Wang, Lingyu Zhang, Qiang Yang, Jieping Ye, and Yan Liu. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. *AAAI*, 2019.

[15] James Douglas Hamilton. *Time series analysis*, volume 2. Princeton university press Princeton, NJ, 1994.

[16] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in neural information processing systems*, pages 1024–1034, 2017.

[17] Minoru Higuchi, Kanji Matsutani, Masahito Kumano, and Masahiro Kimura. Discovering spatio-temporal latent influence in geographical attention dynamics. In *ECML-PKDD*, 2019.

[18] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *CVPR*, 2016.

[19] Weiwei Jiang and Jiayun Luo. Graph neural network for traffic forecasting: A survey. *arXiv preprint arXiv:2101.11174*, 2021.

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[21] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[22] Hema Koppula and Ashutosh Saxena. Learning spatio-temporal structure from rgb-d videos for human activity detection and anticipation. In *ICML*, pages 792–800, 2013.

[23] Ladislav Kristoufek. Measuring correlations between non-stationary series with dcca coefficient. *Physica A: Statistical Mechanics and its Applications*, 402:291–298, 2014.

[24] Nikolay Laptev, Jason Yosinski, Li Erran Li, and Slawek Smyl. Time-series extreme event forecasting with neural networks at uber. In *ICLR*, volume 34, pages 1–5, 2017.

[25] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *ICLR*, 2018.

[26] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

[27] Lei Lin, Zhengbing He, and Srinivas Peeta. Predicting station-level hourly demand in a large-scale bike-sharing network: A graph convolutional neural network approach. *Transportation Research Part C: Emerging Technologies*, 2018.

[28] SPYROS MAKRIDAKIS and MICHÈLE HIBON. Arma models and the box–jenkins methodology. *Journal of Forecasting*, 1997.

[29] Yunqi Miao, Jungong Han, Yongsheng Gao, and Baochang Zhang. St-cnn: Spatial-temporal convolutional neural network for crowd counting in videos. *Pattern Recognition Letters*, 2019.

[30] Yunfei Mu, Jianzhong Wu, Nick Jenkins, Hongjie Jia, and Chengshan Wang. A spatial–temporal model for grid impact analysis of plug-in electric vehicles. *Applied Energy*, 2014.

[31] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*, 2019.

[32] Yuecheng Rong, Zhimian Xu, Ruibo Yan, and Xu Ma. Du-parking: Spatio-temporal big data tells you realtime parking availability. In *SIGKDD*. ACM, 2018.

[33] Ryan A. Rossi. Relational time series forecasting. *KER*, 2018.

[34] Ryan A. Rossi, Di Jin, Sungchul Kim, Nesreen K. Ahmed, Danai Koutra, and John Boaz Lee. From community to role-based graph embeddings. In *arXiv:1908.08572*, 2019.

[35] Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[36] Junkai Sun, Junbo Zhang, Qiaofei Li, Xiuwen Yi, Yuxuan Liang, and Yu Zheng. Predicting citywide crowd flows in irregular regions using multi-view graph convolutional networks. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[37] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

[38] Shanshan Tang, Bo Li, and Haijun Yu. Chebnet: Efficient and stable constructions of deep neural networks with rectified power units using chebyshev approximations. *arXiv preprint arXiv:1911.05467*, 2019.

[39] Waldo Tobler. On the first law of geography: A reply. *Annals of the Association of American Geographers*, 94(2):304–310, 2004.

[40] Bao Wang, Xiyang Luo, Fangbo Zhang, Baichuan Yuan, Andrea L. Bertozzi, and P. Jeffrey Brantingham. Graph-based deep modeling and real time forecasting of sparse spatio-temporal data. *ArXiv*, abs/1804.00684, 2018.

[41] Shuo Wang, Xing Zhang, Jiaxin Zhang, Jian Feng, Wenbo Wang, and Ke Xin. An approach for spatial-temporal traffic modeling in mobile cellular networks. In *27th International Teletraffic Congress*. IEEE, 2015.

[42] Xiaoyang Wang, Yao Ma, Yiqi Wang, Wei Jin, Xin Wang, Jiliang Tang, Caiyan Jia, and Jian Yu. Traffic flow prediction via spatial temporal graph neural network. In *Proceedings of The Web Conference*, 2020.

[43] Tyler Wilson, Pang-Ning Tan, and Lifeng Luo. A low rank weighted graph convolutional approach to weather prediction. In *ICDM*, 2018.

[44] Xian Wu, Yuxiao Dong, Chao Huang, Jian Xu, Dong Wang, and Nitesh V Chawla. Uapd: Predicting urban anomalies from spatial-temporal data. In *ECML-PKDD*, 2017.

[45] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[46] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.

[47] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. In *IJCAI*, 2019.

[48] Na Xu, Pengjian Shang, and Santi Kamae. Modeling traffic flow correlation using dfa and dcca. *Nonlinear Dynamics*, 2010.

[49] Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, and Li Zhenhui. Deep multi-view spatial-temporal network for taxi demand prediction. In *AAAI*, 2018.

[50] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *IJCAI*, 2018.

[51] Qi Zhang, Jianlong Chang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Spatio-temporal graph structure learning for traffic forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1177–1185, 2020.

[52] Qi Zhang, Qizhao Jin, Jianlong Chang, Shiming Xiang, and Chunhong Pan. Kernel-weighted graph convolutional network: A deep learning approach for traffic forecasting. In *ICPR*, pages 1018–1023. IEEE, 2018.