Spatial Network-Wide Traffic Flow Imputation With Graph Neural Network

SUBMISSION DATE / POSTED DATE

03-10-2023 / 05-10-2023

CITATION

Sabzekar, Sina; Bahmani, Rezvan; Ghasemi, Masoud; Amini, Zahra (2023). Spatial Network-Wide Traffic Flow Imputation With Graph Neural Network. TechRxiv. Preprint.
https://doi.org/10.36227/techrxiv.24237181.v1

DOI

10.36227/techrxiv.24237181.v1

Spatial Network-Wide Traffic Flow Imputation With Graph Neural Network

Sina Sabzekar, Rezvan Bahmani, Masoud Ghasemi, Zahra Amini

***Abstract*—Traffic data plays an essential role in Intelligent Transportation Systems (ITS) and offers numerous advantages, including efficient traffic control and system performance improvement. However, due to the scarcity of data collection systems, missing data in traffic datasets is inevitable. Therefore, traffic data imputation becomes an essential task. Graph Neural Network (GNN) is a type of neural network that operates on graph-structured data and have shown potential in handling traffic network related tasks such as traffic prediction and traffic data imputation. In this paper, we contribute to the body of knowledge with two aspects. First, we focus on traffic data imputation using solely spatial information. Most of the studies in the literature address spatio-temporal traffic data imputation, which is a distinct task from our research. Second, Since most GNN models operates on node features, we propose an approach to construct node features for nodes in traffic networks, by leveraging available link flows. To investigate the effectiveness of the proposed method, we implement two missing scenarios, random missing (RM) and block missing (BM). We evaluate the performance of the proposed method on three different sized real-world networks: Sioux Falls, Anaheim, and Chicago. The evaluation results demonstrate that GNN models outperform other baselines for most of the missing patterns.**

***Index Terms*—Intelligent transportation system, traffic data imputation, graph neural network, urban traffic network.**

I. Introduction

Real-time traffic data is crucial for transportation research and practical applications such as traffic control and management, routing, and system performance improvement [1]. Primary sources for collecting traffic data include loop detectors, sensors, cameras, and GPS devices. However, traffic data collection points such as loop detectors and cameras are sparsely deployed within urban areas, resulting in insufficient coverage of the road network in the city. Additionally, communication failure, sensor malfunction, and power outages are common issues faced by these technologies and lead to missing traffic data, which limits downstream applications [2]. Therefore, traffic data imputation becomes a significant task.

Several studies have worked on traffic data imputation over the years. These studies can be classified into three categories. The first category comprises studies that concentrate solely on temporal correlation and disregard spatial dependencies [3], [4]. They mostly focus on averaging values from historical data to impute the missing data. The second category includes studies that rely exclusively on spatial information while assuming temporal information is unavailable [5], [6]. These studies use the information from neighboring data collection points to impute the missing traffic data. The third category encompasses studies that incorporate both temporal and spatial dependencies to enhance imputation accuracy [7], [8]. While it is broadly shown that leveraging both temporal and spatial information is effective for imputing data [9], temporal data analysis is impossible in some cases due to lack of historical data for most locations in the network. Therefore, the need for investigating methods using only spatial information becomes significant.

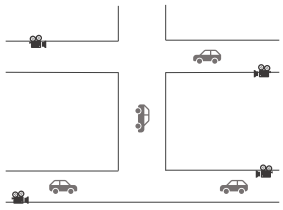


Fig. 1. An example of a section from a traffic network. East-West streets are provided with cameras collecting traffic data while in North-South streets there is a lack of data collecting system.

Spatial traffic data imputation uses the information available from traffic data collection points in addition to information extracted from the network structure to impute the missing traffic data. Fig. 1 shows an example of a section from a traffic network where cameras are provided in East-West streets while in North-South streets there is no data collection system. Using data collected by cameras in East-West streets, the purpose of spatial traffic data imputation is to impute the missing traffic data in North-South streets. There are limited studies in the literature focusing only on spatial information. Early methods leveraged the basic flow conservation law and proposed methods to address the most probable flow estimation problem [10]. Later, methods common in geographical studies such as universal kriging and geographically weighted regression (GWR) were adopted to provide solutions to traffic data imputation [11]. Investigating the methods for spatial traffic data imputation, there were methods based on newly defined centrality indices such as Origin-Destination (OD) centrality [12] and accessibility-weighted centrality [5]. Most recently, machine learning methods have been adopted to address traffic data imputation using spatial information [6], [13].

Machine learning methods have shown their effectiveness in a wide range of tasks, including traffic-related problems such as traffic signal control [14], travel time estimation [15], traffic prediction, and demand forecasting [16]. Recently,

(Corresponding author: Zahra Amini)

S. Sabzekar, M. Ghasemi, and Z. Amini are with the Departement of Civil Engineering, Sharif University of Technology, Tehran, Iran (e-mail: sina.sabzekar@sharif.edu, masoud.ghasemi01@sharif.edu, zahra.amini@sharif.edu)

R. Bahmani is with the School of Electrical and Computer Engineering, College of Engineering, University of Tehran, Tehran, Iran (e-mail: rezvanbahmani@ut.ac.ir)

Graph Neural Networks (GNNs), as a method of machine learning, have shown great potential in traffic network tasks [17], [18] due to their capability in extracting patterns from graph-structured data, which is highly effective for predicting traffic patterns. A road network can be depicted as a graph, with intersections serving as the nodes and roads linking them serving as the edges. GNNs use the available information of nodes known as node features in addition to information related to the neighborhood nodes to generate a representation for that node. This representation is known as node embedding and is used in further tasks such as classification or regression. By using graph of traffic networks as an input information, various GNN models have shown to outperform previous methods in tasks such as predicting road traffic flow and speed [19], [20]. Some notable examples of GNN models include Graph Convolution Networks (GCNs) [21], Diffusion Graph Convolution (DGC) [22], GraphSAGE [23], and Graph Attention Networks (GATs) [24]. Despite promising performance of GNN models, traffic data imputation still faces three significant gaps which considered in this paper.

The first noticeable area that needs attention is that previous studies have predominantly relied on historical data to estimate missing points. However, we believe that incorporating temporal data is often unfeasible in real-world networks due to the limited availability of traffic data for a significant number of nodes and links caused by a shortage of sensors and cameras. Consequently, it becomes imperative to utilize spatial traffic data as the sole source of information to effectively tackle the problem of traffic flow imputation. Secondly, while GNNs have emerged as powerful machine learning models for addressing network-structured datasets, their direct applicability to the problem of flow imputation is limited due to their reliance on node features. Unfortunately, in urban road networks, such node features are often unavailable. To overcome this challenge, this research paper presents a systematic approach that constructs node features using available link data. Thirdly, previous studies have primarily examined traffic data imputation over small subsections of networks with limited sensors [2], [25], [26], while this paper investigates the effectiveness of proposed methods over large-scale real-world networks.

To address these gaps, the paper proposes a novel GNN model that imputes missing traffic flow using only spatial information. The model aggregates flow information from links, transfers it to nodes, and generates node embeddings using a GNN network. These embeddings are then used to estimate missing traffic flows using a Multi-Layer Perceptron (MLP) network. The paper compares the performance of the proposed GNN models with other state-of-the-art machine learning models and conducts experiments on three urban networks of different sizes.

The primary contributions of the paper are:

1) Introducing an approach to transfer available information on graph links to nodes and construct node features.
2) Leveraging a GNN model to interpret the network and find spatial correlations for imputing missing link flows using only spatial information.
3) Implementing two different missing traffic data scenarios (random missing and block missing) and evaluating the proposed method's performance under different rates.
4) Considering the size of the network and proposing a method for large-scale real-world networks.

Overall, the proposed method outperforms other state-of-the-art methods for spatial traffic data imputation, demonstrating robust results across different missing rates and validation datasets.

II. Literature Review

Early Studies. Early traffic imputation methods primarily utilized temporal information, with occasional use of spatial dependencies. One of the earliest methods was Historical Average (HA), which involved averaging values from the identical time periods in the past to impute missing data [3]. Later studies incorporated both temporal and spatial information into their imputation methods. Laña et al. developed a spatial context sensing model that utilizes information from adjacent sensors to impute traffic data [27]. The results of these studies confirmed that utilizing spatial information improves traffic data imputation. Nevertheless, these approaches primarily concentrate on extracting spatial data from neighboring regions and struggle to effectively incorporate network-wide spatial information. Later, tensor factorization techniques have been utilized to represent traffic data as a multi-dimensional tensor, allowing for the imputation of missing data. Article [28] adopted an extended Bayesian probabilistic matrix factorization, while in [29], the authors introduced a temporal factorization framework that merges vector autoregressive process with a low-rank tensor factorization. As opposed to earlier methods, tensor factorization has shown superiority in capturing multidimensional structural dependencies and imputing system-level data. However, its use is restricted to statistical data that has a low rank and requires new learning process for each new batch of incomplete data.

Deep Learning based Methods. Deep learning models have consistently demonstrated their remarkable capacity to uncover hidden patterns from intricate data. Several deep learning models have been employed for traffic engineering tasks, including traffic forecasting and missing traffic data imputation. Duan et al. proposed a denoising stacked autoencoder method, which was the first study that addressed traffic data imputation with deep learning. Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNNs) are two popular approaches for capturing spatio-temporal features of traffic data [30], [31]. Asadi and Regan focused on modeling spatial data using multi-range CNNs and proposed a convolution recurrent autoencoder to solve the problem of missing traffic data imputation [32]. GANs, composed of a generator and a discriminator network, have shown effectiveness in prediction tasks, particularly in traffic prediction [33], [34]. Recently, an attention mechanism has been extensively adopted considering the importance of different features in multi-step traffic prediction [35]. Although CNNs are effective in detecting correlations in data with grid-like structures such as images and RNNs for time series, they lack adequacy

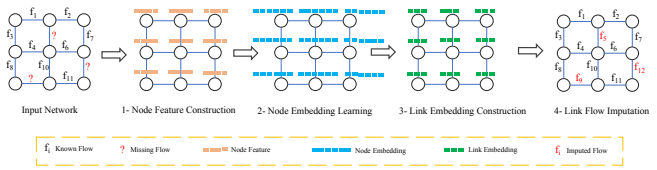


Fig. 2. The overview of the proposed method.

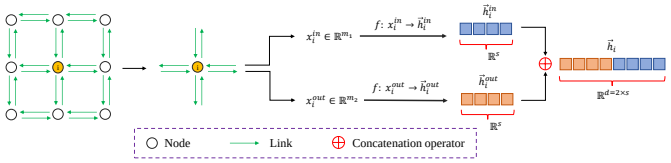


Fig. 3. The overview of the node feature construction module.

in identifying non-linear patterns present in complex road networks.

GNN Methods. Recently, GNN approaches have demonstrated great capability in capturing spatial and temporal features of graph-structured data [36]. In the field of traffic engineering, GNNs have shown remarkable performance in a vast domain of tasks, including travel demand prediction [37], traffic speed prediction [38], traffic accident prediction [39], and vehicle trajectory prediction [40]. Li et al. addressed the traffic forecasting problem by developing a deep learning framework which integrates diffusion graph convolution layers with recurrent architecture [35]. In [41], a model based on GCN was introduced to estimate values for sensors with missing data. To effectively acquire the spatio-temporal relationships from traffic features, attention mechanism [42] and GATs [17] were applied.

III. Preliminaries

In this section, we will demonstrate how to represent the transportation network using a graph-based approach and then we present the formulation of the problem definition.

A. Graph representation of transportation networks

We model the transportation network with a directed graph $G = (V, E, W)$, where V denotes the set of nodes (i.e., intersections), E represents the set of links connecting the nodes (i.e., roads) and W is the set of link weights (i.e., flows on the links). Each link $(i, j) \in E$, has a corresponding weight $w_{ij} \in W$, which reflects the traffic flow on the link. Notably, the links are directed, so $w_{ij} \neq w_{ji}$. To evaluate the performance of our model, we split the dataset into two disjoint subsets: W_T and W_V, representing the training and validation data, respectively. It is important to note that $W_T \cap W_V = \emptyset$ and $W_T \cup W_V = W$, which ensures a clear separation between the data used for training and validation.

B. Problem definition

The objective of traffic flow imputation is to estimate missing flows based on the available values in other links of a network. To achieve this, we leverage available information in the form of $G = (V, E, W_T)$. Our goal is to provide an estimation for W_V, which is denoted by W'_V, indicating imputed flow values. Consequently, traffic flow imputation is defined as a supervised learning task aimed at minimizing the difference between the imputed and actual values. Given that flow values are real numbers, the flow imputation is classified as a regression problem.

IV. Methodology

In this section, we extensively discuss our proposed model for traffic flow imputation. Our approach consists of four distinct modules, namely (1) node feature construction, (2) node embedding learning, (3) link embedding construction, and (4) link flow imputation, as illustrated in Fig. 2. In the subsequent sections, we provide a comprehensive elaboration of each module.

A. Node feature construction

Most GNN models use node features as input and generate node embeddings by aggregating the node feature with information obtained from adjacent nodes. This technique is referred to as message passing, which enables information propagation between nodes in the graph, allowing GNNs to capture complex dependencies and patterns in data. The generated embeddings can then be used for various tasks such as node classification or link prediction. However, our problem here lacks the necessary node features since link flows are the only available information, making it impossible to apply GNNs directly to the nodes. To address this issue, in the first step, we construct node features from link flows. By utilizing the available information of link flows in the neighborhood of a node, we construct a feature vector for that node. This feature vector can then be used to impute the missing flows of other links connecting to the node. Relatively speaking, if $\overrightarrow{h_i} \in \mathbb{R}^d$ and $\overrightarrow{h_j} \in \mathbb{R}^d$ represent the constructed node features of nodes i and j respectively, where d is the number of features per each node. Hence, $\overrightarrow{h_i}$ and $\overrightarrow{h_j}$ can be used to estimate w_{ij}, the flow value for the link connecting node i to node j.

We propose a novel method for constructing node feature vectors that utilizes available information about links, as shown in Fig. 3. Through this method, first, each node is isolated and the link flows coming to or going out of the node are considered. Next, two sets x_i^{in} and x_i^{out} are created, which include the incoming and outgoing link flows, respectively. These sets are defined in equations (1) and (2) and can have different sizes (m_1 and m_2). Then, we leverage a function f to project these sets into vectors $\overrightarrow{h_i}^{in}$ and $\overrightarrow{h_i}^{out}$. Finally, these vectors are concatenated to construct the vector $\overrightarrow{h_i}$, which represents the constructed features for node i.

$$x_i^{in} = \{w_{ji} : i, j \in V, j \in N(i), w_{ji} \in W_T\} \quad (1)$$

$$x_i^{out} = \{w_{ij} : i, j \in V, j \in N(i), w_{ij} \in W_T\} \quad (2)$$

Where $N(i)$ represents the nodes in neighborhood of node i. The challenge we encounter is to propose the function f that satisfies two essential conditions. Firstly, the function must support inputs of varying sizes due to the unknown number

TABLE I
VARIABLES USED IN FEATURE VECTOR OF NODES.

Variable	Definition	Variable	Definition
$f_i^{sum,in}$	$\sum_{a \in x_i^{in}} a$	$f_i^{sum,out}$	$\sum_{a \in x_i^{out}} a$
$f_i^{mean,in}$	$\frac{1}{\|x_i^{in}\|} \sum_{a \in x_i^{in}} a$	$f_i^{mean,out}$	$\frac{1}{\|x_i^{out}\|} \sum_{a \in x_i^{out}} a$
$f_i^{min,in}$	$\min_{a \in x_i^{in}}(a)$	$f_i^{min,out}$	$\min_{a \in x_i^{out}}(a)$
$f_i^{max,in}$	$\max_{a \in x_i^{in}}(a)$	$f_i^{max,out}$	$\max_{a \in x_i^{out}}(a)$

$\|x_i^{in}\|$ and $\|x_i^{out}\|$ indicate the cardinality (i.e., number of elements) of the sets x_i^{in} and x_i^{out}, respectively. in and out represent the inflow and outflow, respectively.

of links in each node's neighborhood. Secondly, it must be permutation invariant, meaning that the same results should be obtained regardless of how the inputs are ordered. Hence, we propose to employ four basic mathematical operators that satisfy the conditions of function f, including sum, $mean$, min, and max. Each of these operators serves a specific purpose. sum represents flow equilibrium in the node, $mean$ indicates distribution of flows in links connected to the node, min denotes minimum flow passing through the node, and max provides information related to the capacity of the node. In addition to traffic flow information that these four operators provide, they support inputs of varying size and exhibit permutation-invariant property.

We use the introduced operators to construct node feature vector $\overrightarrow{h_i}$, which is defined as follows:

$$\overrightarrow{h_i} = (f_i^{sum,in}, f_i^{mean,in}, f_i^{min,in}, f_i^{max,in}, f_i^{sum,out}, f_i^{mean,out}, f_i^{min,out}, f_i^{max,out}) \tag{3}$$

Where each variable in $\overrightarrow{h_i}$ is defined in Table I.

B. Node Embeddings

In the previous section, we discussed the process for constructing node features. This section presents methods for generating node embeddings, which are representations of nodes that depend on both the structure of the network and available node features. The purpose of these methods is to encode nodes as low-dimensional vectors summarizing their position in graph and local graph neighborhood structure [43]. For node u, we denote its embedding with z_u where $z_u \in \mathbb{R}^{d'}$.

GNNs aim to generate node embeddings through the process called message passing. There are different types of GNNs including GraphSAGE, GCN, and GAT, which differ in how they generate embeddings. In this paper, we focus on two popular GNN models, GCN and GAT.

Generally, GNNs take in a set of node features, $\boldsymbol{h} = \{\overrightarrow{h_1}, \overrightarrow{h_2}, ..., \overrightarrow{h_N}\}, \overrightarrow{h_i} \in \mathbb{R}^d$, where N is the number of nodes, and d represents the number of features per each node. The layer generates a set of new features, $\boldsymbol{h'} = \{\overrightarrow{h'_1}, \overrightarrow{h'_2}, \ldots, \overrightarrow{h'_N}\}, \overrightarrow{h'_i} \in \mathbb{R}^{d'}$, where d' can differ from d. The output is an embedding for each node obtained from either equation (4) or (5) for GCN and GAT models, respectively. Equation (4) shows that GCN leverages node degrees to average the neighbors' information, while GAT employs an attention coefficient for averaging, as indicated in equation (5).

$$\overrightarrow{h'_i} = \sigma\left(\mathbf{W} \sum_{j \in N(i) \cup i} \frac{\overrightarrow{h_i}}{\sqrt{\|N(i))\| \, \|N(j))\|}}\right) \tag{4}$$

$$\overrightarrow{h'_i} = \sigma\left(\sum_{j \in N(i)} \alpha_{ij} \mathbf{W} \overrightarrow{h_i}\right) \tag{5}$$

Where σ is a non-linear function, α_{ij} is an attention coefficient that determines the importance of node j's features to node i, calculated based on equation (6), and $\mathbf{W}$ is a learnable linear projection matrix ($\mathbf{W} \in \mathbb{R}^{d' \times d}$) that converts input features into higher-level features.

$$\alpha_{ij} = \frac{exp\left(LeakyReLU(\overrightarrow{a}^T \left[\mathbf{W}\overrightarrow{h_i} \parallel \mathbf{W}\overrightarrow{h_j}\right])\right)}{\sum_{k \in N(i)} exp\left(LeakyReLU(\overrightarrow{a}^T \left[\mathbf{W}\overrightarrow{h_i} \parallel \mathbf{W}\overrightarrow{h_j}\right]\right)} \tag{6}$$

Where $\overrightarrow{a}$ is a learnable vector of parameters ($\overrightarrow{a} \in \mathbb{R}^{2 \times d'}$), $\overrightarrow{a}^T$ represents the transpose of the vector $\overrightarrow{a}$, and $\parallel$ indicates the concatenation operator.

In this study, we implement both GCN and GAT in our proposed method to generate node embeddings, and compare their performance."

C. Link Embeddings

In section IV-B, GNN models are utilized to generate node embeddings. In this section, we are aiming to use these node embeddings to generate link embeddings. Similar to node embeddings, link embeddings are vectors that represent each link in the network, including the information of the nodes at both ends of the link, as well as the position of the link within the network.

One widely used approach for constructing link embeddings involves concatenating the embeddings of the two connected nodes [43]. Mathematically, let $\overrightarrow{h'_i}$ and $\overrightarrow{h'_j}$ be the embeddings for nodes i and j, respectively. We can concatenate these embeddings to obtain a link embedding $\overrightarrow{e}_{ij}$. We define the set of link embeddings, denoted by $\boldsymbol{e}$, to include the embeddings for all links in the training set, and it is defined as follows:

$$\boldsymbol{e} = \{\overrightarrow{e}_{ij} : i, j \in V, w_{ij} \in W_T\} \tag{7}$$

Where $\overrightarrow{e}_{ij} \in \mathbb{R}^{2 \times d'}$ and is defined as:

$$\overrightarrow{e}_{ij} = \left[\overrightarrow{h'_i} \parallel \overrightarrow{h'_j}\right] \tag{8}$$

Where $\parallel$ denotes concatenation operator. The set of link embeddings is further used in link imputation task, which is explained in following section.

D. Missing Link Flow Imputation

In section IV-C, we discussed how to generate link embeddings. This section aims to use link embeddings to impute flow values of links. To achieve this, we propose the use of a Multi-Layer Perceptron (MLP) network that maps link

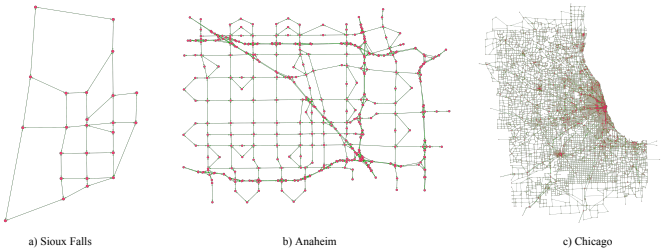


Fig. 4. Visualization of test networks. The scales are different among these networks.

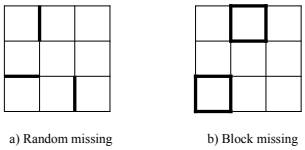


Fig. 5. Types of missing pattern. Missing links are denoted by thick lines.

embeddings to link flows. This can be represented by the following equation:

$$W'_V = \sigma(\mathbf{W} \times \boldsymbol{e}) \tag{9}$$

Where W'_V denotes imputed link flows as defined previously, σ refers to an activation function, $\mathbf{W}$ is the weight matrix of the MLP network, and $\boldsymbol{e}$ is the set of link embeddings.

E. Performance Metrics

This section introduces three performance metrics used to evaluate the proposed method's effectiveness. These metrics include Pearson Correlation Coefficient (PCC), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE), which are defined as follows:

$$\text{PCC} = \frac{1}{n-1}\sum_{k=1}^{n}\left(\frac{\hat{y}_i - \hat{y}}{s_{\hat{y}}}\right)\left(\frac{y_i - \bar{y}}{s_y}\right) \tag{10}$$

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}} \tag{11}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i| \tag{12}$$

Where n is the number of links in validation set (i.e., $\|W_T\|$), $\hat{y}$ represents the imputed flow values, $\bar{\hat{y}}$ and $s_{\hat{y}}$ indicate the mean and standard deviation of $\hat{y}$ respectively. Similarly, y represents the actual flow values, $\bar{y}$ and s_y indicate the mean and standard deviation of y, respectively. Higher absolute values of PCC indicate higher correlations between imputed and actual values. Lower values of RMSE and MAE indicate lower errors in flow imputation.

V. Experiments

A. Data description

In this study, three real world networks are used to test the performance of the proposed method along with the other popular methods. Test cases are Sioux Falls, Anaheim, and Chicago with corresponding network sizes of small, medium, and large, as shown in Fig. 4. These networks are provided by Transportation Networks for Research Project [44] and commonly used for traffic assignment problems in which the flows on links are known. Table II displays the characteristics of these networks, including the number of nodes and links, average degree, and other network's characteristics. Table III shows the characteristic of the traffic flow in these networks. It is notable that the traffic flows in each of the three test networks are normalized within the range of [0, 1].

TABLE II
Networks characteristics.

Dataset	Size	# Nodes	# Links	Average degree
Sioux Falls	Small	24	76	3.166
Anaheim	Medium	416	914	2.197
Chicago	Large	12979	39018	3.006

TABLE III
Flow characteristic in test networks.

Dataset	Min	Max	Mean
Sioux Falls	4494.658	23192.283	11547.409
Anaheim	0.0	13602.2	2009.962
Chicago	0.0	18007.0	815.709

B. Missing pattern generation

This section discusses two types of missing patterns that commonly occur in traffic networks and discusses the approach used to generate them. The first type is RM, which occurs when missing traffic flow records are randomly distributed throughout the network with no correlation between them (Fig. 5 (a)). The second type is BM, which occurs when values are spatially dependent. A block contains several adjacent links that are correlated with each other (Fig. 5 (b)).

To generate RM, we split the links randomly into a train and validation set. The missing rate, defined as the ratio of links in the validation set to the total number of links, is used to measure the percentage of missing data. The formula for missing rate is as follows:

$$\text{Missing rate} = \frac{\|W_V\|}{\|W\|} \tag{13}$$

Implementing the BM scenario involves two steps. First, we identify all the blocks in the network. Second, some of these blocks are randomly selected as missing blocks, and all links within these blocks are considered missing. A block is a cycle of links, typically contains four links. In this study, we identify all blocks consisting of three to six links. Table IV provides details about BM for different datasets. The number of blocks in each dataset is set manually to achieve similar missing rate.

C. Baselines

To evaluate the efficacy of the proposed method, we compare GNNs with the following baselines:

1) KNN (K-Nearest Neighbors) [45]: KNN uses the average of traffic flow from neighborhood of missing traffic

TABLE IV
DETAILS OF BM SCENARIO.

Dataset	# Total Blocks	# Missing Blocks	# Missing Links	# Missing Rate
Sioux Falls	11	2	14	0.184
Anaheim	88	20	132	0.144
Chicago	3766	1000	7598	0.195

Note: Number of links is averaged over the replications.

data point as the imputation, which mainly focuses on spatial correlations of traffic flow.

2) SVR (Support Vector Regression) [46]: SVR finds the nonlinear dependency between the existing traffic feature and traffic flow using a linear function in high dimensional feature space.
3) FCNN (Fully Connected Neural Network) [47]: FCNN also known as Multi-Layer Perceptron (MLP) works by taking input data, passing it through a series of layers of interconnected neurons, and using gradient descent to adjust the weights of the connections between neurons in order to minimize an error function and improve the accuracy of the traffic flow imputation.

To ensure a fair comparison between the proposed GNN models and baselines, we utilized identical node feature vectors across all models.

D. Experiment settings

The experiments are conducted on an Intel i7 6500U CPU. Data is randomly split into train and validation sets. To prevent overfitting, a fixed number of epochs (500) are set for all deep-learning models (FCNN, GCN, and GAT), and an early stopping strategy is adopted. Additionally, Adam is used as the optimizer for all deep-learning models, with the learning rate set to 0.001 for FCNN, 0.01 for GCN, and 0.01 for GAT. Furthermore, the weight decay is kept constant at 0.0005, and the mean squared error (MSE) loss function is employed across all models. These parameters are determined through trial and error and proved to obtain best performance.

For the SVR, we utilize Grid search to tune hyperparameters. We select the penalty parameter (C) and the parameter defining the influence of a single training record ($Gamma$) from the sets $\{0.1, 1, 10, 100, 1000\}$, and $\{1, 0.1, 0.01, 0.001\}$, respectively. In the case of the KNN, we explore various values for the parameter k through grid search, and it is chosen from set $\{1, 2, ..., 10\}$.

The FCNN consists of two linear layers having ReLU activation functions. The dimensions of the layers are (16,8) and (8,1) correspondingly. A dropout layer with a probability of 0.2 is applied as a regularization layer. The GCN and GAT both have two layers with dimensions of (8,32) and (32,16). The same architecture is employed for the Multi-Layer Perceptron (MLP) network, which is utilized for link weight prediction in both GCN and GAT models. It has the same dimensions as the FCNN network, make the comparison between the models more reliable.

VI. RESULTS

A. Imputation performance analysis

In this section, we compare the performance of GNN models with the baselines for different missing patterns and three datasets. The evaluation has been conducted for different missing rates. Each experiment has been repeated for ten times and results are averaged to mitigate the influence of random numbers. Results are shown through the Tables V-VII. Overall, GNN models are superior to other baseline models in most scenarios. Both GNN models perform well and the difference between them is marginal.

For the Sioux Falls dataset, KNN almost performs better than other methods by a considerable margin for most missing patterns. In missing rates 0.5 and 0.8, although the KNN achieves better results based on RMSE and MAE metrics, PCCs show that GNN models have better performance. For the Anaheim and Chicago datasets, GNN models are superior to baseline models for all missing patterns and missing ratios. Both GNN models show similarly competitive performance and their difference is marginal. For BM, GCN outperforms other methods. On Anaheim network, GCN is slightly performs better than GAT but on Chicago network, GAT is marginally superior. This result completely meets our expectation that GAT excels on more complex networks than GCN. Overall, the results indicate that GNN models can provide more accurate and robust results than existing methods for diverse combinations of missing patterns and different urban networks. This is surely related to the ability of GNN models in capturing spatial dependencies in graph-structured data. It should be noted that on both Anaheim and Chicago datasets, GCN slightly outperform GAT when the missing ratio reaches 0.8 or when the missing pattern is BM. It is evident that the attention mechanism adversely affects the model's performance in more complicated missing scenarios leading to slightly weaker performance of GAT in comparison to GCN. The reason why GNN models perform less accurately on Sioux Falls dataset in comparison to KNN is probably due to characteristics of Sioux Falls network. It is a small-sized synthetic network with limited number of nodes leading the GNN models to generate identical node embeddings during the message passing process and resulting in less competitive performance of these models.

B. Analysis on imputation residuals

In this section, we conduct an analysis of the imputation residuals, which serve as a crucial indicator of the model's performance. The residual is computed by subtracting the actual flow values from the imputed flow values, represented as $\hat{y} - y$. Positive and negative residuals signify overestimation and underestimation of traffic flows, respectively. To visually present the residuals of both GNN models (GCN and GAT) and the baselines for the Chicago network at a missing rate of 0.5, we provide histograms in Fig. 6. Our findings reveal that the GNN models predominantly exhibit overestimation of flow values, while the baselines tend to underestimate them. When using traffic data for scheduling and planning, overestimation is preferred to underestimation. This preference stems from the

TABLE V
IMPUTATION PERFORMANCE ON SIOUX FALLS DATASET.

Sioux Falls												
Model	Missing rate = 0.2			Missing rate = 0.5			Missing rate = 0.8			Block missing		
	RMSE	MAE	PCC	RMSE	MAE	PCC	RMSE	MAE	PCC	RMSE	MAE	PCC
KNN	**0.222**	**0.187**	0.624	**0.294**	**0.225**	0.198	**0.351**	**0.264**	0.158	**0.304**	**0.242**	**0.018**
SVR	0.232	0.191	**0.654**	0.349	0.277	0.127	0.439	0.350	0.120	0.364	0.300	0.007
FCNN	0.251	0.207	0.574	0.342	0.267	0.021	0.399	0.308	0.099	0.358	0.293	0.002
GCN	0.257	0.225	0.612	0.317	0.247	0.212	0.399	0.311	**0.191**	0.338	0.266	0.013
GAT	0.280	0.240	0.601	0.305	0.231	**0.227**	0.431	0.329	0.103	0.338	0.272	0.011

The bold values represent the best imputation results for each metric.

TABLE VI
IMPUTATION PERFORMANCE ON ANAHEIM DATASET.

Anaheim												
Model	Missing rate = 0.2			Missing rate = 0.5			Missing rate = 0.8			Block missing		
	RMSE	MAE	PCC	RMSE	MAE	PCC	RMSE	MAE	PCC	RMSE	MAE	PCC
KNN	0.206	0.115	0.376	0.220	0.130	0.247	0.264	0.160	0.103	0.269	0.181	0.053
SVR	0.252	0.159	0.284	0.294	0.183	0.197	0.338	0.244	0.076	0.361	0.340	0.012
FCNN	0.214	0.136	0.353	0.256	0.159	0.231	0.286	0.184	0.097	0.344	0.239	0.033
GCN	**0.167**	**0.108**	**0.574**	**0.176**	**0.113**	0.415	**0.240**	**0.145**	0.120	**0.258**	**0.176**	**0.083**
GAT	0.172	**0.108**	0.541	0.177	**0.113**	**0.424**	0.242	0.146	**0.141**	0.275	0.184	0.076

The bold values represent the best imputation results for each metric.

TABLE VII
IMPUTATION PERFORMANCE ON CHICAGO DATASET.

Chicago												
Model	Missing rate = 0.2			Missing rate = 0.5			Missing rate = 0.8			Block missing		
	RMSE	MAE	PCC	RMSE	MAE	PCC	RMSE	MAE	PCC	RMSE	MAE	PCC
KNN	0.074	0.034	0.339	0.078	0.034	0.255	0.089	0.042	0.145	0.050	0.031	0.387
SVR	0.129	0.062	0.094	0.124	0.071	0.003	0.156	0.117	0.012	0.106	0.055	0.016
FCNN	0.087	0.040	0.122	0.083	0.039	0.146	0.106	0.065	0.047	0.071	0.041	0.174
GCN	0.068	0.035	0.539	0.066	**0.032**	0.503	**0.076**	**0.034**	0.372	**0.046**	**0.028**	**0.392**
GAT	**0.063**	**0.032**	**0.556**	**0.065**	**0.032**	**0.527**	**0.076**	**0.034**	**0.375**	0.049	0.031	0.384

The bold values represent the best imputation results for each metric.

fact that overestimation facilitates planning for more critical scenarios and contributes to the development of more reliable systems.

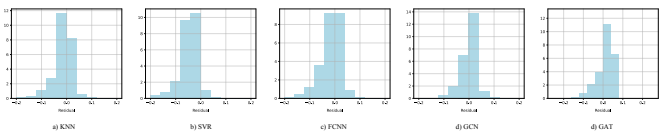


Fig. 6. Imputation residuals of GNN models and baselines tested on Chicago network when the missing rate is 0.5.

C. Analysis on RM and BM patterns

This section presents an analysis of the performance of the proposed GNN models in two distinct scenarios. Fig. 7 illustrates two sections from the Chicago network: Fig. 7 (a) represents a BM scenario with two missing blocks, where all links within these blocks have missing flows while the flows of other links in this section are known. Fig. 7 (b) shows a RM scenario, where links with missing flows are randomly distributed. To evaluate the imputed flow values for these two scenarios, we refer to Table VIII. In the RM scenario, we selected two links, namely 12301 $\rightarrow$ 5846 and 11912 $\rightarrow$ 5865, which have actual flow values of 395 and 620, respectively. Notably, GAT outperforms other methods in accurately estimating the flow values for both of these links, closely approximating the actual values. Additionally, GCN exhibits superior performance compared to the baselines in the RM scenario, albeit slightly less accurate than GAT. In the BM scenario, we choose two links, 10602 $\rightarrow$ 6999 and 6973 $\rightarrow$ 6978 - with actual flow values of 829 and 571, respectively. For the link 10602 $\rightarrow$ 6999, GAT demonstrates accurate performance, providing an imputed flow value that

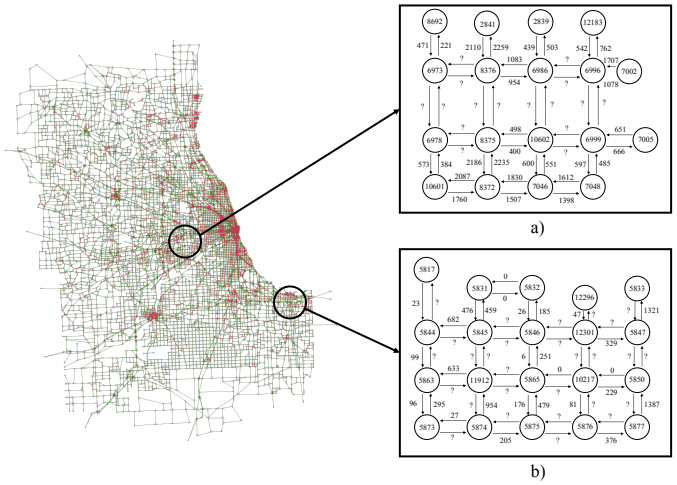


Fig. 7. Visualization of the a) BM and b) RM scenarios in Chicago network. Missing rate is 0.5 in RM Scenario.

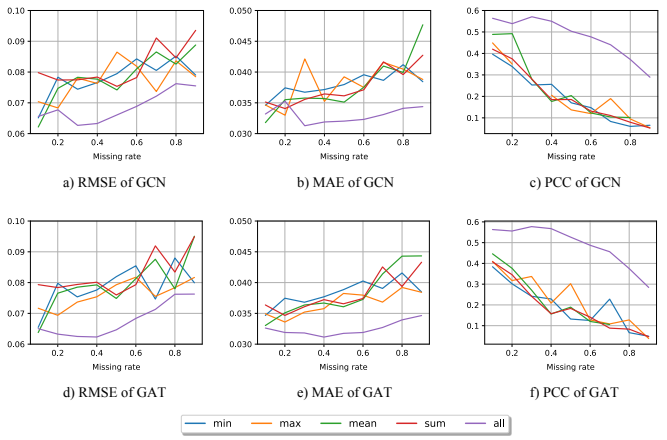


Fig. 8. Comparison of the performance of the node feature construction operators.

closely aligns with the actual flow. On the other hand, for the link 6973 $\rightarrow$ 6978, GCN marginally surpasses GAT and the other baselines in terms of accuracy. It is worth mentioning that GAT tends to overestimate the actual flows, while the other methods tend to underestimate them (with the exception of GCN for 11912 $\rightarrow$ 5865). This discrepancy aligns with the desirable behavior in traffic-related tasks, as discussed in detail in Section VI-B. Overall, the results presented in the table indicate that despite the complexity of the BM scenario, GNN models are capable of learning the hidden dependencies and demonstrating effectiveness comparable to that observed in the RM scenario.

D. Analysis on node construction operators

In Section IV-A, we discussed the node construction process, wherein link flow information is utilized to generate feature vectors for nodes. Our approach employed four operators: min, max, $mean$, and sum. This section delves into an investigation of the individual effectiveness of these operators on GNN models' performances in traffic flow imputation when used exclusively. Fig. 8 illustrates the comparative results of this analysis. Additionally, we present the performance of GNN models when all four operators are combined. The figure demonstrates that each of the four operators, when employed independently, produces almost indistinguishable performance outcomes for both GCN and GAT models. Notably, the figure provides compelling evidence that utilizing all four operators simultaneously significantly improves the performance of both GCN and GAT models across all performance metrics and under nearly all missing rates.

E. Analysis on Computation cost

In this section, we explore the computational efficiency of GNN models in terms of time and memory. We also include FCNN as a baseline model in this comparison. Table IX displays the results. The training time is given per each epoch of training due to different epochs in each model. Inference time is the whole time needed for the model to predict on the test set. The RAM used is an indicator of the memory cost by models in training. It is shown that FCNN is the most efficient considering both the time and memory used. However, the imputation results of FCNN are disappointing as discussed in section VI-A. The GNN models have far higher time-complexities in comparison to FCNN which is due to message passing process which is computationally complex. The training time of GAT is almost twice the GCN's which is caused by the attention mechanism involved in GAT. The inference time for GCN and GAT is almost the same. The RAM used by three methods is marginally different which is not significant. Although GAT is slightly better than GCN in complex networks, but its training time is much larger than GCN's. These findings indicate that GCN demonstrates comparable imputation performance to GAT, while requiring significantly less computational resources.

VII. Conclusion

This study focuses on the problem of traffic flow imputation. While most of the studies in the literature have addressed the issue by employing spatio-temporal information, we used only spatial information. We propose to adopt two GNN models, GCN and GAT, to address the traffic flow imputation problem. However, applying GNNs to this problem presents a challenge because they rely heavily on node features that are absent in the given data. Moreover, the available information is limited to link flows. To overcome this challenge, the authors propose a node feature construction module, which generates a vector of features for each node based on link flows in its vicinity. The proposed method consists of three other modules: Node Embeddings, Link Embeddings, and Missing Link Flow Imputation. These modules leverage the constructed node features to represent nodes and links and then map these representations to link flows. To evaluate the performance of the proposed methods, we implement two missing scenarios, RM and BM, using three real-world traffic networks, namely Sioux Falls, Anaheim, and Chicago. The evaluation results indicate that GNN models outperform other baselines in different missing scenarios and missing rates.

TABLE VIII
IMPUTED FLOWS OF PROPOSED MODELS AND BASELINES IN RM AND BM SCENARIOS.

Scenario		RM		BM	
Link		123010 → 5846	11912 → 5865	10602 → 6999	6973 → 6978
Actual flow		395	620	829	571
Imputed flow	KNN	289	394	509	322
	SVR	218	179	923	688
	FCNN	176	288	643	292
	GCN	303	823	922	594
	GAT	406	792	835	627

All values are reported in veh/hour. Bold values denote the most accurate imputed values among the models.

TABLE IX
COMPUTATION COST OF NEURAL NETWORK-BASED MODELS ON CHICAGO DATASET WHEN THE MISSING RATE IS 0.5.

Model	Training time (ms) / epoch	Inference time (ms)	RAM used (MB)
FCNN	5.88	4.52	560.94
GCN	43.22	20.01	568.28
GAT	87.93	20.02	570.26

Times are reported in milliseconds (ms) for more readability.

Furthermore, the authors investigate the effectiveness of the four operators used in the node feature construction module. The findings show that each of the four operators performs similarly, but their contributions together can improve the model's performance.

REFERENCES

[1] Q. Wang, C. Guo, H.-N. Dai, and M. Xia, "Variant-depth neural networks for deblurring traffic images in intelligent transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[2] Y. Liang, Z. Zhao, and L. Sun, "Memory-augmented dynamic graph convolution networks for traffic data imputation with diverse missing patterns," *Transportation Research Part C: Emerging Technologies*, vol. 143, p. 103826, 2022.

[3] B. L. Smith, W. T. Scherer, and J. H. Conklin, "Exploring imputation techniques for missing data in transportation management systems," *Transportation Research Record*, vol. 1836, no. 1, pp. 132–142, 2003.

[4] M. Zhong, P. Lingras, and S. Sharma, "Estimation of missing traffic counts using factor, genetic, neural, and regression techniques," *Transportation Research Part C: Emerging Technologies*, vol. 12, no. 2, pp. 139–166, 2004.

[5] G. Sarlas and K. W. Axhausen, "Exploring spatial methods for prediction of traffic volumes," in *16th Swiss Transport Research Conference (STRC 2016)*. Swiss Transport Research Conference (STRC), Conference Proceedings.

[6] X. Yao, Y. Gao, D. Zhu, E. Manley, J. Wang, and Y. Liu, "Spatial origin-destination flow imputation using graph convolutional networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 12, pp. 7474–7484, 2020.

[7] 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, Conference Proceedings, pp. 890–897.

[8] Y. Chen, K. Li, C. K. Yeo, and K. Li, "Traffic forecasting with graph spatial–temporal position recurrent network," *Neural Networks*, vol. 162, pp. 340–349, 2023.

[9] T. Sun, S. Zhu, R. Hao, B. Sun, and J. Xie, "Traffic missing data imputation: A selective overview of temporal theories and algorithms," *Mathematics*, vol. 10, no. 14, p. 2544, 2022.

[10] R. Zohar and D. Geiger, "Estimation of flows in flow networks," *European journal of operational research*, vol. 176, no. 2, pp. 691–706, 2007.

[11] B. Selby and K. M. Kockelman, "Spatial prediction of traffic levels in unmeasured locations: applications of universal kriging and geographically weighted regression," *Journal of Transport Geography*, vol. 29, pp. 24–32, 2013.

[12] M. Lowry, "Spatial interpolation of traffic counts based on origin–destination centrality," *Journal of Transport Geography*, vol. 36, pp. 98–105, 2014.

[13] D. Xu, C. Wei, P. Peng, Q. Xuan, and H. Guo, "Ge-gan: A novel deep learning framework for road traffic state estimation," *Transportation Research Part C: Emerging Technologies*, vol. 117, p. 102635, 2020.

[14] D. Ma, B. Zhou, X. Song, and H. Dai, "A deep reinforcement learning approach to traffic signal control with temporal traffic pattern mining," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 11 789–11 800, 2021.

[15] W. Zhou, X. Xiao, Y.-J. Gong, J. Chen, J. Fang, N. Tan, N. Ma, Q. Li, C. Hua, and S.-W. Jeon, "Travel time distribution estimation by learning representations over temporal attributed graphs," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[16] H. Xu, T. Zou, M. Liu, Y. Qiao, J. Wang, and X. Li, "Adaptive spatiotemporal dependence learning for multi-mode transportation demand prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 18 632–18 642, 2022.

[17] Y. Ye, S. Zhang, and J. J. Yu, "Spatial-temporal traffic data imputation via graph attention convolutional network," in *Artificial Neural Networks and Machine Learning–ICANN 2021: 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 14–17, 2021, Proceedings, Part I*. Springer, Conference Proceedings, pp. 241–252.

[18] D. Xu, X. Shang, H. Peng, and H. Li, "Mvhgn: Multi-view adaptive hierarchical spatial graph convolution network based trajectory prediction for heterogeneous traffic-agents," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[19] S. Fukuda, H. Uchida, H. Fujii, and T. Yamada, "Short-term prediction of traffic flow under incident conditions using graph convolutional recurrent neural network and traffic simulation," *IET Intelligent Transport Systems*, vol. 14, no. 8, pp. 936–946, 2020.

[20] D. Feng, Z. Wu, J. Zhang, and Z. Wu, "Dynamic global-local spatial-temporal network for traffic speed prediction," *IEEE Access*, vol. 8, pp. 209 296–209 307, 2020.

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

[22] J. Atwood and D. Towsley, "Diffusion-convolutional neural networks," *Advances in neural information processing systems*, vol. 29, 2016.

[23] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[24] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[25] Z. Cui, L. Lin, Z. Pu, and Y. Wang, "Graph markov network for traffic forecasting with missing data," *Transportation Research Part C: Emerging Technologies*, vol. 117, p. 102671, 2020.

[26] W. Zhang, F. Zhu, Y. Lv, C. Tan, W. Liu, X. Zhang, and F.-Y. Wang, "Adapgl: An adaptive graph learning algorithm for traffic prediction based on spatiotemporal neural networks," *Transportation Research Part C: Emerging Technologies*, vol. 139, p. 103659, 2022.

[27] I. Laña, I. I. Olabarrieta, M. Vélez, and J. Del Ser, "On the imputation of missing data for road traffic forecasting: New insights and novel techniques," *Transportation research part C: emerging technologies*, vol. 90, pp. 18–33, 2018.

[28] X. Chen, Z. He, and L. Sun, "A bayesian tensor decomposition approach for spatiotemporal traffic data imputation," *Transportation research part C: emerging technologies*, vol. 98, pp. 73–84, 2019.

[29] X. Chen and L. Sun, "Bayesian temporal factorization for multidimensional time series prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 4659–4673, 2021.

[30] Y. Duan, Y. Lv, Y.-L. Liu, and F.-Y. Wang, "An efficient realization of deep learning for traffic data imputation," *Transportation research part C: emerging technologies*, vol. 72, pp. 168–181, 2016.

[31] X. Dai, R. Fu, E. Zhao, Z. Zhang, Y. Lin, F.-Y. Wang, and L. Li, "Deeptrend 2.0: A light-weighted multi-scale traffic prediction model using detrending," *Transportation Research Part C: Emerging Technologies*, vol. 103, pp. 142–157, 2019.

[32] R. Asadi and A. Regan, "A convolution recurrent autoencoder for spatio-temporal missing data imputation," *arXiv preprint arXiv:1904.12413*, 2019.

[33] Y. Chen, Y. Lv, X. Wang, and F.-Y. Wang, "Traffic flow prediction with parallel data," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, Conference Proceedings, pp. 614–619.

[34] Y. Lv, Y. Chen, L. Li, and F.-Y. Wang, "Generative adversarial networks for parallel transportation systems," *IEEE Intelligent Transportation Systems Magazine*, vol. 10, no. 3, pp. 4–10, 2018.

[35] Y. Cui, B. Jin, F. Zhang, and X. Sun, "A deep spatio-temporal attention-based neural network for passenger flow prediction," in *Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, Conference Proceedings, pp. 20–30.

[36] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, Conference Proceedings, pp. 753–763.

[37] Y. Liang, G. Huang, and Z. Zhao, "Bike sharing demand prediction based on knowledge sharing across modes: A graph-based deep learning approach," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, Conference Proceedings, pp. 857–862.

[38] 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.

[39] L. Yu, B. Du, X. Hu, L. Sun, L. Han, and W. Lv, "Deep spatio-temporal graph convolutional network for traffic accident prediction," *Neurocomputing*, vol. 423, pp. 135–147, 2021.

[40] Y. Liang and Z. Zhao, "Nettraj: A network-based vehicle trajectory prediction model with directional representation and spatiotemporal attention mechanisms," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 14 470–14 481, 2021.

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

[42] C. Zheng, X. Fan, C. Wang, and J. Qi, "Gman: A graph multi-attention network for traffic prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, Conference Proceedings, pp. 1234–1241.

[43] W. L. Hamilton, "Graph representation learning. morgan & claypool publishers," 2020.

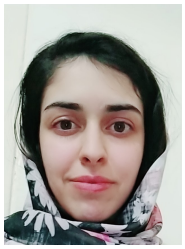
[44] T. N. f. R. C. Team, "Transportation networks for research." [Online]. Available: https://github.com/bstabler/TransportationNetworks

[45] H. Samet, *The design and analysis of spatial data structures*. Addison-wesley Reading, MA, 1990, vol. 85.

[46] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, Conference Proceedings, pp. 144–152.

[47] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

Sina Sabzekar obtained his bachelor's degree in Civil Engineering with a minor in Computer Engineering from the University of Tehran in 2020. Currently, he is pursuing a Master of Science degree in Transportation Engineering at Sharif University of Technology. With a keen interest in intelligent transportation systems and machine learning applications, his research revolves around addressing transportation problems through innovative approaches.



Rezvan Bahmani obtained her bachelor's degree in Computer Engineering from the University of Tehran in 2023. Her research interests include machine learning and its applications.

Masoud Ghasemi is the lead of the AI team at Snapp, the largest ride-hailing platform in Iran. With a passion for leveraging technology to optimize transportation systems, he has dedicated his career to developing AI-related software solutions. Currently pursuing a Ph.D. in Transportation Planning at Sharif University of Technology.

Zahra Amini is currently an Assistant Professor at the Department of Civil Engineering, Sharif University of Technology. She completed her bachelor's degree in 2014 and her master's degree in 2015, in Civil Engineering at University of California, Berkeley. She obtained her Ph.D. in Highway and Traffic Engineering, in 2018 at University of California, Berkeley. Her research interests are Intelligent Transportation System (ITS), traffic theory and control strategies, and transportation system operation and management.