Multistage Graph Convolutional Network With Spatial Attention for Multivariate Time Series Imputation

Qianyi Chen, Jiannong Cao, *Fellow, IEEE*, Yu Yang, Wanyu Lin, *Member, IEEE*, Sumei Wang, and Youwu Wang

***Abstract*— In multivariate time series (MTS) analysis, data loss is a critical issue that degrades analytical model performance and impairs downstream tasks such as structural health monitoring (SHM) and traffic flow monitoring. In real-world applications, MTS is usually collected by multiple types of sensors, making MTS and correlations between variates heterogeneous. However, existing MTS imputation methods overlook the heterogeneous correlations by manipulating heterogeneous MTS as a homogeneous entity, leading to inaccurate imputation results. Besides, correlations between different data types vary due to ever-changing environmental conditions, forming dynamic correlations in MTS. How to properly learn the hidden correlation from heterogeneous MTS for accurate data imputation remains unresolved. To solve the problem, we propose a multistage graph convolutional network with spatial attention (MSA-GCN). In the first stage, we decompose heterogeneous MTS into several clusters with homogeneous data collected from identical sensor types and learn intracluster correlations. Then, we devise a GCN with spatial attention to explore dynamic intercluster correlations, which is the second stage of MSA-GCN. In the last stage, we decode the learned features from previous stages via stacked convolutional neural networks. We jointly train these three-stage models to predict the missing data in MTS. Leveraging this multistage architecture and spatial attention mechanism makes MSA-GCN effectively learn heterogeneous and dynamic correlations among MTS, resulting in superior imputation performance. We tested MSA-GCN with the monitoring data from a large-span bridge and Wetterstation weather dataset. The results affirm its superiority over baseline models, demonstrating its enhanced accuracy in reducing imputation errors across diverse datasets.**

***Index Terms*— Data imputation, graph convolutional network (GCN), structural health monitoring (SHM).**

Received 25 May 2023; revised 31 May 2024; accepted 13 October 2024. Date of publication 6 November 2024; date of current version 9 July 2025. This work was supported in part by Shenzhen–Hong Kong–Macau Technology Research Program Type C under Grant SGDX20201103095203029; in part by HK RGC Theme-Based Research Scheme under Grant PolyU T43-513/23-N and Grant T22-502/18R; in part by the Research Institute for Artificial Intelligence of Things, The Hong Kong Polytechnic University; and in part by the Innovation and Technology Commission of Hong Kong SAR Government, China, under Grant K-BBY1. *(Corresponding author: Yu Yang.)*

Qianyi Chen is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, and also with the School of Engineering, Westlake University, China (e-mail: chenqianyi@westlake.edu.cn).

Jiannong Cao and Wanyu Lin are with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong (e-mail: csjcao@comp.polyu.edu.hk; wanylin@comp.polyu.edu.hk).

Yu Yang is with the Centre for Learning, Teaching and Technology, The Education University of Hong Kong, Hong Kong (e-mail: yangyy@eduhk.hk).

Sumei Wang and Youwu Wang are with the Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University, Hong Kong (e-mail: may.sm.wang@polyu.edu.hk; yw.wang@connect.polyu.hk).

Digital Object Identifier 10.1109/TNNLS.2024.3486349

I. Introduction

DATA loss issues frequently occur in real-world data acquisition and transmission due to unavoidable incidents, such as sensor malfunctions and noise interference. These problems pose challenges for time series analysis methods, resulting in decreased performance for many related downstream applications, including structural health monitoring (SHM), traffic flow monitoring, and a wide range of Internet-of-Things (IoT) applications. Consequently, data imputation, also known as data completion or data reconstruction, has been extensively researched in the literature. Existing data imputation solutions generally fall into two categories: model-based (correlation-based) methods and data-driven methods. Traditional model-based approaches involve modeling temporal correlations within one variable or spatial correlations between multiple variables with explicit mathematical equations that come from domain-specific numerical models. The missing values are recovered based on those correlations and available valid data. Model-based methods have been extensively studied in IoT systems for engineering structures [1], [2], transportation system [3], [4], smart manufacturing [5], and air quality monitoring [6], [7]. A typical example is recovering the missing data for an SHM system using correlations derived from finite element models for engineering structures.

Implementing model-based methods in real-world IoT systems can be challenging for several reasons. First, a gap inevitably exists between predefined equations and the actual IoT system. Correlations between real-world measurement data are often more elusive, as variables are subject to numerous unexpected factors or hidden variables. However, the "white-box" models [8] utilized in model-based methods are typically constructed based on manual assumptions and simplifications, which lead to suboptimal performance when implemented. Second, explicit equations that describe correlations between variables are unavailable in many complex IoT systems. Despite the availability of massive amounts of data, many interactions between these variables still remain undiscovered and elusive. These challenges restrict the practical application of model-based solutions in engineering practice.

Data-driven methods refer to methods that work directly from data and do not require domain-specific models. As a promising alternative, data-driven methods can automatically

learn correlations between variables and are usually applicable across various fields and applications. Derived from measurement data, these methods can simulate real-world systems with higher fidelity. The rapid advancement of data-driven data imputation methods has been facilitated by the unprecedented availability of high-fidelity measurements from real-world IoT systems, numerical simulations, and experimental data. Traditional data-driven models include matrix factorization [9], autoregressive integrated moving average models [10], and k-means [11]. Meanwhile, progress in machine learning (ML) algorithms has provided researchers with a wealth of statistical models for data imputation [12], including convolutional neural networks (CNNs), recurrent neural networks (RNNs), graph neural networks (GNNs), and generative adversarial networks (GANs).

However, large-scale IoT systems typically consist of multiple monitoring items, resulting in heterogeneous measurement variates. The monitoring item here refers to a variable measured by a group of sensors deployed in different locations of a monitoring system. It is different from the sensor type because the same type of sensor can be used for different monitoring items. For instance, accelerometers on the bridge deck and accelerometers on cables belong to different monitoring items although they are all accelerometers. They describe the vibration of different parts of a bridge, exhibiting distinct amplitude and fluctuation patterns.

Heterogeneous measurement variates lead to heterogeneous correlations between variates, encompassing correlations within the same monitoring items and between different monitoring items. Furthermore, correlations between different monitoring items are inherently dynamic because they are influenced by ever-changing environmental conditions and other unexpected factors. These heterogeneous and dynamic correlations bring difficulties for existing data-driven methods to learn well.

For instance, an SHM system for a bridge includes various monitoring items, such as the acceleration of the bridge deck and cables, displacement of piers, and multipoint temperature. Acceleration, temperature, and displacement data have distinct modalities, e.g., sampling frequencies, amplitudes, and fluctuation patterns. Therefore, the correlations between data within each item naturally exhibit different patterns. Furthermore, the correlation between bridge deck acceleration and cable acceleration is dynamic, exhibiting significant variations based on external conditions. Under strong wind conditions, these two variables are closely correlated. However, their correlation diminishes under traffic load, which only excites a small region of the bridge.

Existing data-driven methods often overlook heterogeneous and dynamic correlations, impairing data imputation accuracy. Most of these methods are designed for a single monitoring item, such as acceleration data [13], [14], [15], strain [16], velocity [17], displacement [18], and temperature [19]. Some other methods were proposed for data imputation while monitoring multiple items. Nevertheless, they either manipulate heterogeneous data as a homogeneous entity [20] or only use data from one item to optimize the imputation model's parameters for another monitoring item [21]. None of the existing methods simultaneously explore the intercorrelation and intracorrelation among the monitoring items.

We propose a multistage graph convolutional network with spatial attention (MSA-GCN) for multivariate time series (MTS) imputation. Specifically, in the first stage, we decompose the sensors of an IoT system into multiple sensor clusters based on their monitoring items. We apply multiple self-designed intracluster encoders within each sensor cluster to capture the spatial correlations between homogeneous sensors and temporal dependencies between different time steps, respectively. In the second stage, we adopt an advanced GCN with spatial attention to learn the dynamic interactions between different monitoring items. In the third stage, data for each sensor are recovered using features extracted in the previous two stages. We jointly train these three-stage models and predict missing values in heterogeneous MTS. The innovative architecture helps MSA-GCN to explore heterogeneous and dynamic correlations in heterogeneous data efficiently.

The contributions of this article are highlighted as follows.

1) We propose MSA-GCN, an innovative multistage ML model, to solve a new data loss issue. In this issue, data loss happens in heterogeneous MTS collected for multiple monitoring items. The multistage architecture enables MSA-GCN to divide heterogeneous data correlations into two levels: correlations within each monitoring item and between multiple monitoring items. Afterward, MSA-GCN can incorporate different networks dedicated to capturing the two levels of correlations, respectively, leading to proper feature learning.
2) In MSA-GCN, we design "n" encoders in the first stage to independently learn data correlations within each monitoring item. This approach enables MSA-GCN to achieve higher feature extraction accuracy for heterogeneous data, as data collected in different monitoring items inherently exhibit distinct interactions.
3) MSA-GCN incorporates a spatial attention mechanism in the second stage to accurately learn the dynamic correlations between different monitoring items. Spatial attention is calculated based on the output from the first stage and subsequently used to dynamically adjust the weights of the graph's edges in the second stage. MSA-GCN can effectively capture dynamic correlations by assigning different weights to edges according to the strength of the correlations.
4) We utilize two datasets to evaluate the accuracy of our proposed MSA-GCN. One is a real-world monitoring dataset from Tsing Ma Bridge in Hong Kong and the other is the Wetterstation weather dataset. Our findings reveal that MSA-GCN outperforms existing data imputation baseline models in terms of imputation accuracy and demonstrates robustness under varying data loss severities.

Section II presents related works and the distinctions to our method. Problem formulation is presented in Section III. In Section IV, we illustrate the technical details of the MSA-GCN model. Experimental results are presented in Section V before we conclude the article and outline future directions in Section VI.

II. Related Work

MTS imputation involves restoring missing values in incomplete time series data collected from multiple sensors. These time series can be either homogeneous or heterogeneous. Homogeneous time series pertain to data collected by the same type of sensor for the same monitoring item, while heterogeneous data encompass data from multiple monitoring items. In this section, we survey existing ML-based imputation methods for dealing with homogeneous and heterogeneous MTS.

A. Data Imputation for Homogeneous MTS

Numerous IoT systems measure a single monitoring item at multiple locations, resulting in homogeneous data collection, such as air pollution data from various air quality monitoring stations within a city [22], energy consumption data from an enterprise collected by multiple smart meters [23], and traffic data at multiple intelligent transportation systems units [24]. Researchers have developed diverse ML models to uncover hidden correlations within the data and perform data imputation. Early works primarily utilized CNNs [13], [15] to extract spatial correlations. Many studies also employed RNNs [25], [26] to exploit temporal dependencies in time series. Recently, there has been growing interest in GNN as they are inherently designed to learn from graph-shaped data, which can naturally describe sensor networks and irregular spatial correlations [27], [28]. In addition, various GANs [16], [18], [29], [30] and their variants have been proposed for data imputation tasks. GANs enable data-driven models to accurately mimic the original distribution of data, and GAN-based data imputation solutions have demonstrated remarkable performance in certain datasets.

Some researchers have combined different types of neural networks to extract more comprehensive features. Yu et al. [6] integrated CNN and GAN to recover air pollution data, while Liang et al. [24] combined temporal convolution network (TCN) and GNN to effectively and efficiently extract temporal and spatial features for accurate data recovery. Wu et al. [7] proposed an ML model that integrated bi-directional RNN (BRNN), denoising auto-encoder (DAE), and GAN to recover air quality monitoring data. Nevertheless, real-world IoT systems always consist of multiple monitoring items and generate heterogeneous data. None of the above-mentioned approaches are designed to extract correlations from heterogeneous data and they might fail to recover the missing values of heterogeneous data correctly.

B. Data Imputation for Heterogeneous MTS

An increasing number of IoT systems have begun to collect heterogeneous data using various types of sensors for more comprehensive sensing. Although leveraging these heterogeneous data appropriately helps uncover hidden correlations and enables more accurate data imputation. It is more challenging than data imputation with homogeneous data. Because heterogeneous data typically exhibit significant differences in characteristics. These data have different sizes, amplitudes, and fluctuation patterns and their correlations are also heterogeneous.

Nevertheless, few existing data imputation studies have focused on heterogeneous data. Niu et al. [21] developed a spatiotemporal graph attention network to restore missing cable forces using valid cable force data and temperature data. However, this method cannot simultaneously impute missing temperature data, as temperature data are not used as node attributes in the graph but rather used to compute attention coefficients. As a result, the correlations between multiple temperature measurement points and between temperature and cable forces are not explored. Hou et al. [31] proposed an RNN-based method for data imputation between different data types. However, they mix strain and displacement data to generate a whole entity as the input. Since no hint is given to the RNN to distinguish between strain and displacement data, this method cannot effectively learn correlations between heterogeneous data. In contrast, the MSA-GCN proposed in this article is designed to investigate both correlations within each monitoring item and between different monitoring items, enabling simultaneous prediction of missing values from multiple monitoring items.

III. Problem Formulation

A. Notations and Data Loss Model

In an IoT system, we define a heterogeneous MTS as a collection of MTS from n sensor clusters. Each sensor cluster contains multiple sensors that generate time series for the same monitoring item, such as temperature, displacement, and wind velocity. We assume that time series from different clusters exhibit heterogeneity, which can manifest as differences in sampling frequency or the number of sensors across clusters. Data within the same sensor cluster are homogeneous.

We denote the raw data collected from the ith cluster within a period of time as $\mathbf{F_i^{input}} \in \mathbb{R}^{S_i \times T_i}, i = 1, \ldots, n$, where S_i denotes the number of sensors in this cluster and T_i denotes the number of sampling steps in this monitored period. Data loss can happen in any sensor and we represent it using n masks for n sensor clusters, respectively. The ith mask is denoted as $\mathbf{M_i} \in \mathbb{R}^{S_i \times T_i}, i = 1, \ldots, n$. The value in the sth row and tth column in $\mathbf{M_i}$ is denoted as m_{st}. It represents that corresponding value of sensor s in time step t is missing ($m_{st} = 0$) or valid ($m_{st} = 1$) in $\mathbf{F_i^{input}}$. In this article, variables denoted with bold letters represent multidimensional arrays, and variables denoted with italic letters represent single values.

Fig. 1 illustrates the data imputation task for the incomplete heterogeneous time series. In this figure, we use NAN, which means *Not A Number*, to represent each missing value. In practical IoT systems, measurement data can be lost in a single sampling step or across consecutive sampling steps. To properly simulate real-world data loss conditions, we use two data-missing scenarios simultaneously when generating datasets. We adopt the *block missing* data loss setting from GRIN [32]. Specifically, there is a failure at each sampling step with a probability of p_{block}. The duration of each node failure is uniformly sampled from $\min_{\text{step}}$ to $\max_{\text{step}}$. In addition, we apply the point missing to the remaining valid data, by removing 5% of them randomly.

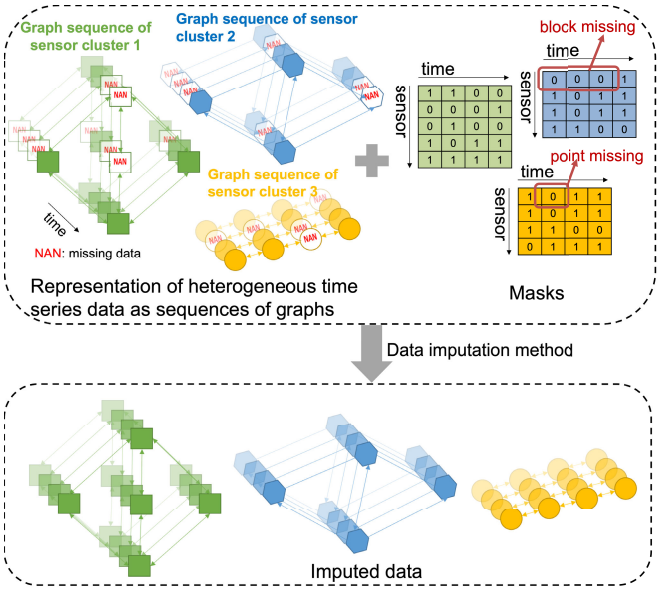


Fig. 1. Illustration of data imputation task for heterogeneous MTS.

B. Data Imputation Problem for Heterogeneous MTS

We formulate the data imputation problem for heterogeneous MTS as follows.

1) *Given:* A heterogenous dataset $\{\mathbf{F}_1^{\text{input}}, \ldots, \mathbf{F}_n^{\text{input}}\}$ consists of data from n sensor clusters and a group of corresponding masks $\{\mathbf{M}_1, \ldots, \mathbf{M}_n\}$ that represent data loss.
2) *Assumption:* Variates from the same sensor cluster are homogeneous and variates from different sensor clusters are heterogeneous. Data loss happens in every variate.
3) *Objective:* Train a data-driven model to recover the missing values in the heterogeneous dataset.

The well-trained model is used to recover the missing data for multiple sensor clusters simultaneously shown in (1) and (2)

$$\left\{\hat{\mathbf{Y}}_1, \ldots, \hat{\mathbf{Y}}_n\right\} = f\left(\left\{\mathbf{F}_1^{\text{input}}, \ldots, \mathbf{F}_n^{\text{input}}\right\}, \{\mathbf{M}_1, \ldots, \mathbf{M}_n\}\right) \quad (1)$$

where $\hat{\mathbf{Y}}_i \in \mathbb{R}^{S_i \times T_i}, i = 1, \ldots, n$ denotes the predicted value for the ith cluster. f denotes the well-trained data imputation model. Afterward, $\hat{\mathbf{Y}}_i$ are filled into the raw data $\mathbf{F}_i^{\text{input}}$ to generate the imputed data $\mathbf{Y}_i$ for the ith sensor cluster using

$$\begin{aligned}\mathbf{Y}_i &= \mathbf{Impute}\left(\mathbf{F}_i^{\text{input}}, \hat{\mathbf{Y}}_i, \mathbf{M}_i\right) \\ &= \mathbf{F}_i^{\text{input}} \odot \mathbf{M}_i + \hat{\mathbf{Y}}_i \odot (\mathbf{1} - \mathbf{M}_i)\end{aligned} \quad (2)$$

where the symbol $\odot$ denotes Hadamard product.

IV. Methodology

A. Overall Framework

Considering the differences between intercluster correlations and intracluster correlations in heterogeneous data, it can be improper to learn these correlations with one GNN. As a result, we design MSA-GCN to learn them separately. The overall architecture is illustrated in Fig. 2. The proposed MSA-GCN consists of three stages: the first two stages are used for encoding raw data, while the last stage works for decoding. Specifically, there are n separate self-designed spatial-temporal encoders in stage 1, with each encoding the graph sequence from one sensor cluster independently. In stage 2, each vertex of the input graph represents features extracted from each sensor cluster in stage 1. We incorporate an attention-based GNN in stage 2 to explore the intercluster correlations. The spatial attention mechanism is suitable for the diverse and dynamic intercluster correlations between different sensor clusters, as it can dynamically adjust the strength of correlations between vertices in each training step. Stage 3 decodes the feature from stage 2 and receives local features from stage 1 through a residual connection. It fuses the two sets of features and makes the final prediction.

B. Stage 1: Intracluster Encoder

As shown in Fig. 2, the intracluster encoder comprises n encoders, which operate with n sensor clusters, respectively. Fig. 3 displays the details of components in one encoder. In each encoder, two spatial-temporal encoders encode data sequences in forward and backward directions, respectively, along the time axis. Their outputs are first merged with a direction merge encoder (DME). Subsequently, the outputs of DME are upsampled along the time axis and merged along the channel axis with a channel merge encoder (CME). Afterward, the output of each intracluster encoder in stage 1 has the same shape and will serve as the vertex feature in the input graph of stage 2.

We propose TD-GRIN, a temporal decayed GRIN module, as the spatial-temporal encoders in stage 1. GRIN [32] is a state-of-the-art spatial-temporal encoder for MTS imputation. We innovatively integrate it with temporal decayed factors because sequential data are often missing in consecutive time steps. The decayed factor δ^d [25], defined in (3), helps describe the time gap between the current time step s_t and the last observation. It provides a hint for GRIN to update hidden states accurately based on the hidden states in the last time step, together with the prediction for the current time step

$$\delta_t^d = \begin{cases} 1 + \delta_{t-1}^d, & \text{if } t > 1, m_{st} = 0 \\ 1, & \text{if } t > 1, m_{st} = 1 \\ 0, & \text{if } t = 1. \end{cases} \quad (3)$$

Algorithm 1 illustrates the forward pass of a TD-GRIN. We slice $\mathbf{F}^{\text{input}}$ along the time axis and obtain $\{\mathbf{f}_1^{\text{input}}, \mathbf{f}_2^{\text{input}}, \ldots, \mathbf{f}_T^{\text{input}}\}$, where $\mathbf{f}_t^{\text{input}} \in \mathbb{R}^S, t = 1, \ldots, T$. The same operation is applied to mask $\mathbf{M}$ and we obtain $\{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_T\}$. $\mathbf{h}$ denotes the hidden state in TD-GRIN. $\mathbf{A}^{\text{adj}}$ denotes the adjacent matrix for one sensor cluster. **Revs** denotes the operation that reverses a list along time direction. **Conv1d** denotes a 1-D CNN. **SpatialDecoder** comprises linear readout layers and one message-passing neural network. Please refer to [32] for details on the spatial decoder. $\mathbf{TD-MPGRU}$ is a self-designed recurrent message passing network cell that updates the hidden state $\mathbf{h}$. It will be introduced later.

Similar to the GRIN module, we obtain preliminary imputation results $\mathbf{F}^{\text{pred1}}$, $\mathbf{F}^{\text{pred2}}$, and $\mathbf{F}^{\text{repr}}$ as the representation for

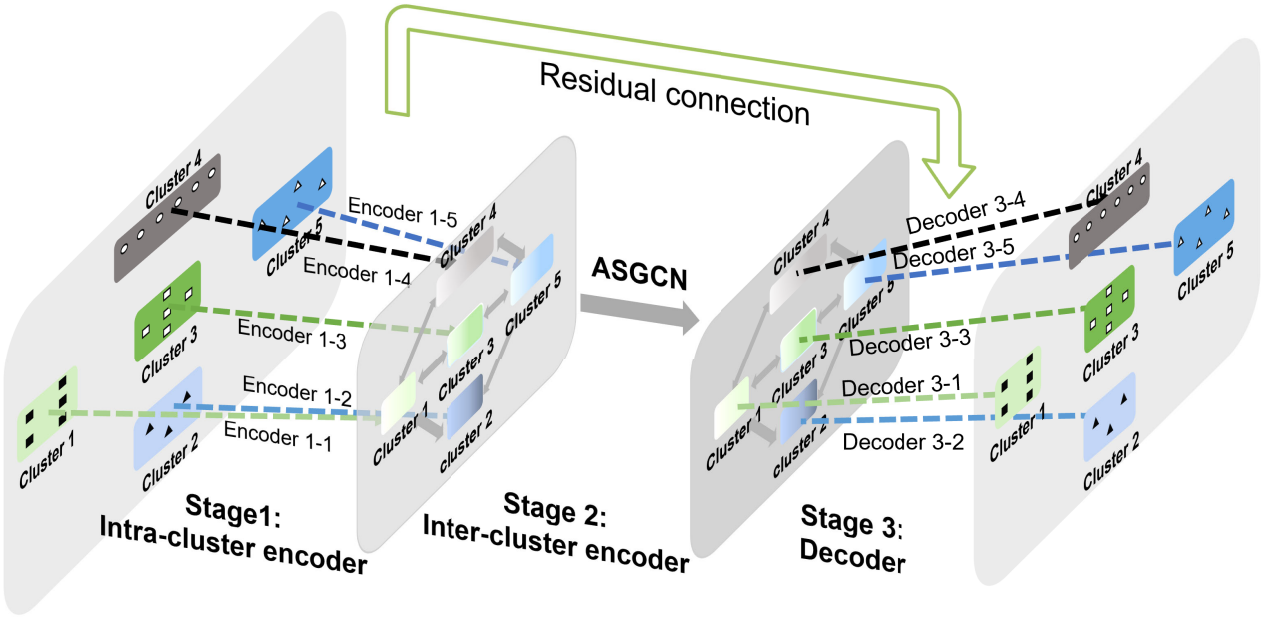


Fig. 2. Overall architecture of MSA-GCN. The input heterogeneous data are divided into five clusters and MSA-GCN consists of three stages. Stages 1 and 3 have multiple networks to encode and decode data for each sensor cluster. The number of clusters in MSA-GCN is not fixed at 5. It varies with different datasets.

intracluster features extracted by TD-GRIN. Since we have GRIN modules for two directions, we obtain four preliminary imputation results with the same size as the input $\mathbf{F}^{\text{input}}$. They are stacked into $\mathbf{F}^{\text{imp}} \in \mathbb{R}^{4\times S\times T}$, denoting the imputation in stage 1. In the ith encoder in stage 1, intracluster features are concatenated with the mask following:

$$\mathbf{F}_i^{\text{intra}} = \mathbf{F}_i^{\text{repr,fwd}} \Big|\Big| \mathbf{F}_i^{\text{repr,bwd}} \Big|\Big| \mathbf{M_i} \tag{4}$$

where $\mathbf{F}_i^{\text{intra}} \in \mathbb{R}^{(2\times H_i^{\text{TD-GRIN}}+1)\times S_i\times T_i}$, $i = 1, \ldots, n$. $H_i^{\text{TD-GRIN}}$ is the size of hidden states in TD-GRIN. $\mathbf{F}_i^{\text{repr,fwd}}$ and $\mathbf{F}_i^{\text{repr,bwd}}$ denote the TD-GRIN extracted intracluster features for ith sensor cluster in forward and backward directions, respectively. || denotes a concatenation operator. $\mathbf{F}_i^{\text{intra}}$ contains both spatial and temporal features from the ith sensor cluster.

The forward pass of $\mathbf{TD-MPGRU}$ is detailed in Algorithm 2. Its input comprises the current hidden state $\mathbf{h}$, $\mathbf{x}_t$, mask $\mathbf{m}_t$, and decay factor δ^d for the current time step, and the adjacency matrix $\mathbf{A}_{\text{adj}}$. In contrast to **MPGRU** in [32], we incorporate the decay factor δ^d when computing the update state $\mathbf{u}_t$. In Algorithm 2, δ^d is first transformed via a linear transformation **NN**. We then obtain the temporal decay factor γ_t by encoding it further with a 1-D CNN. The estimated hidden state $\mathbf{h}^\delta$ and estimated feature $\hat{\mathbf{x}}_t^\delta$ are generated with γ_t and $\mathbf{h}$. Here, **MPNN** denotes the message-passing neural network, while σ and **tanh** represent the sigmoid and tanh activation functions, respectively.

Our proposed DME consists of two 2-D CNNs, which process $\mathbf{F}_i^{\text{intra}}$ to encode its $(2 \times H_i^{\text{TD-GRIN}} + 1)$ channels into a single channel. Considering the fact that different sensor clusters have varying numbers of sensors, S_i and T_i can differ among clusters. However, to generate a graph-shaped input for stage 2, n intracluster features $\mathbf{F}_i^{\text{intra}}, i = 1, \ldots, n$ should be encoded into tensors of identical shape. To accomplish this transformation, we introduce the CME. In the CME, we first use an up-sampling layer to standardize time sequences collected at different sampling frequencies to a uniform size. Subsequently, a 1-D convolution layer is employed to merge features along the channel axis, which corresponds to sensors in the sensor cluster. The merged feature at this stage is

Algorithm 1 TD-GRIN

Input: $\{\mathbf{f}_1^{input}, \mathbf{f}_2^{input}, \ldots, \mathbf{f}_T^{input}\}, \{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_T\}, \delta^d, \mathbf{h}, \mathbf{A}^{adj}$, Direction = Forward or Backward

```
if Direction = Backward
    F^input = Revs(F^input) = {f_T^input, ..., f_2^input, f_1^input}
    δ^d = Revs(δ^d)
    M = Revs(M)
end
for t = 1 → T do
    ŷ_t^pred1 = Conv1d(h)
    x_t = m_t ⊙ f_t^input + (1 − m_t) ⊙ ŷ_t^pred1
    ŷ_t^pred2, f_t^repr = SpatialDecoder(x_t, m_t, h, A^adj)
    x_t = m_t ⊙ f_t^input + (1 − m_t) ⊙ ŷ_t^pred2
    h = TD − MPGRU(h, x_t, m_t, A^adj, δ_t^d)
end
Ŷ^pred1 = {ŷ_1^pred1, ŷ_2^pred1, ..., ŷ_T^pred1}
Ŷ^pred2 = {ŷ_1^pred2, ŷ_2^pred2, ..., ŷ_T^pred2}
F^repr = {f_1^repr, f_2^repr, ..., f_T^repr}
if Direction = Backward
    Ŷ^pred1 = Revs(Ŷ^pred1)
    Ŷ^pred2 = Revs(Ŷ^pred2)
    F^repr = Revs(F^repr)
```

Output: $\hat{\mathbf{Y}}^{pred1}, \hat{\mathbf{Y}}^{pred2}, \mathbf{F}^{repr}$

Algorithm 2 TD-MPGRU

Input: $\mathbf{h}, \mathbf{x}_t, \mathbf{m}_t, \mathbf{A}^{adj}, \delta_t^d$

$\gamma_t = \mathbf{exp}\{-\mathbf{Relu}(\mathbf{NN}(\delta_t^d))\}$
$\gamma_t = \mathbf{Conv1d}(\gamma_t)$
$\mathbf{h}^{\delta} = \mathbf{h} \odot \gamma_t$
$\hat{\mathbf{x}}_t^{\delta} = \mathbf{NN}(\mathbf{h}^{\delta})$
$\mathbf{x}_t^{\delta} = \mathbf{Impute}(\hat{\mathbf{x}}_t^{\delta}, \mathbf{x}_t, \mathbf{m}_t)$
$\mathbf{r_t} = \sigma\{\mathbf{MPNN}(\mathbf{x}_t||\mathbf{m}_t||\mathbf{h}, \mathbf{A}^{adj})\}$
$\mathbf{u_t} = \sigma\{\mathbf{MPNN}(\mathbf{x}_t^{\delta}||\mathbf{m}_t||\mathbf{h}^{\delta}, \mathbf{A}^{adj})\}$
$\mathbf{c_t} = \mathbf{tanh}\{\mathbf{MPNN}(\mathbf{x}_t^{\delta}||\mathbf{m}_t||\mathbf{r}_t \odot \mathbf{h}^{\delta}, \mathbf{A}^{adj})\}$
$\mathbf{h} = \mathbf{u}_t \odot \mathbf{h} + (1 - \mathbf{u}_t) \odot \mathbf{c}_t$

Output: $\mathbf{h}$

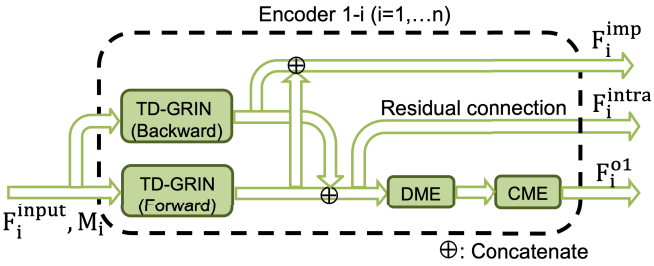


Fig. 3. Architecture of one encoder in stage 1. It make preliminary prediction $\mathbf{F}^{\text{imp}}$, extract intracluster feature $\mathbf{F}^{\text{intra}}$, and merge features into $\mathbf{F}_i^{O1}$ as input for stage 2.

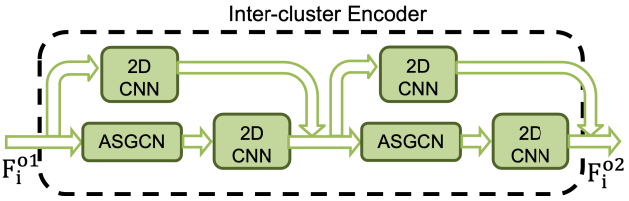


Fig. 4. Architecture of one encoder in stage 2. It extracts intercluster features and encode them into $\mathbf{F}_i^{O2}$ as the input for stage 3.

denoted as $\mathbf{F}_i^{O1}$. Regardless of the varying sensor counts and sampling frequencies across different sensor clusters, $\mathbf{F}_i^{O1}$ from each sensor cluster maintains a consistent shape. These features are then used as an input graph for the second stage.

C. Stage 2: Intercluster Encoder

Fig. 4 illustrates the intercluster encoder utilized in stage 2. It is composed of two concatenated repeated modules. In each module, attention-based spatial graph convolutional networks (ASGCNs) [33] extract intercluster features followed by one 2-D CNN. Another 2-D CNN works as a residual connection. The output of this stage is $\mathbf{F}_i^{O2}$, representing the encoded intercluster features as the input for stage 3. The 2-D CNNs, being widely used feature extractors, are adept at extracting spatial features. In this section, we primarily introduce ASGCNs, which are employed to explore the diverse and dynamic correlations between different sensor clusters.

The traditional graph convolutional network (GCN) is employed to extract spatial dependencies from the input graph sequence. The forward pass in a GCN can be represented as follows:

$$\begin{aligned}\mathbf{F}^{l+1} &= \mathbf{GCN}\left(\mathbf{F}^{l}, \mathbf{A}^{\text{adj}}\right) \\ &= \sigma\left(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}^{\text{adj}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{F}^{l}\mathbf{W}^{l} + \mathbf{B}^{l}\right)\end{aligned} \tag{5}$$

where $\mathbf{F}^{(l)} \in \mathbb{R}^{S\times H^l}$ is the input feature in layer l and $\mathbf{F}^{(l+1)} \in \mathbb{R}^{S\times H^{l+1}}$ is the output of the GCN. $\tilde{\mathbf{A}}^{\text{adj}} = \mathbf{A}^{\text{adj}} + \mathbf{I}_n$ where $\mathbf{A}^{\text{adj}}$ is the adjacent matrix of the input graph. Degree matrix $\tilde{\mathbf{D}}$ is a diagonal matrix with element in the diagonal as $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}^{\text{adj}}$. $\mathbf{W}^{(l)}$ and $\mathbf{B}^{(l)}$ are learnable parameters. σ is a nonlinear activation function.

In ASGCN, the spatial attention matrix $\mathbf{S}$ is computed with

$$\mathbf{S} = \mathbf{V}^s \cdot \sigma\left(\left(\mathbf{F}^{o1}\mathbf{W}_1\right)\left(\mathbf{W}_2\mathbf{F}^{o1}\right)^T + \mathbf{b}^s\right) \tag{6}$$

where $\mathbf{V}^s \in \mathbb{R}^{n\times n}, \mathbf{W}_1 \in \mathbb{R}^{h_{o1}}, \mathbf{W}_2 \in \mathbb{R}^{h_{o1}}$ are learnable weights of ASGCN. $\mathbf{b}^s$ are learnable bias. σ denotes the activation function. The value $\mathbf{S_{ij}}$ in $\mathbf{S}$ represents the strength of correlation between vertex i and j. It is updated during each training step with the new coming input data. A softmax function shown in (7) is then applied to $\mathbf{S}$ to ensure the sum of weights to be 1. Then, we got $\mathbf{S}'$ for adjusting the adjacent matrix in GCN

$$\mathbf{S}'_{i,j} = \frac{\mathbf{exp}\left(\mathbf{S}_{\text{i,j}}\right)}{\sum_{j=1}^{N}\mathbf{exp}\left(\mathbf{S}_{\text{i,j}}\right)}. \tag{7}$$

The computed spatial attention matrix will be used to adjust the terms of the Chebyshev polynomial in the Chebyshev polynomial approximation of the graph convolution operation. We use graph Laplacian matrix $\mathbf{L}$ as the graph filter and g_θ as the kernel. The graph convolution operation is defined as

$$\mathbf{F}^{l+1} = g_\theta(\mathbf{L})\mathbf{F}^{l} = g_\theta\left(\mathbf{U}\Lambda\mathbf{U}^T\right)\mathbf{F}^{l} = \mathbf{U}g_\theta(\Lambda)\mathbf{U}^T\mathbf{F}^{l} \tag{8}$$

where $\mathbf{L} = \mathbf{I_N} - \mathbf{D}^{-(1/2)}\mathbf{A}^{\text{adj}}\mathbf{D}^{-(1/2)}$. $\mathbf{A}^{\text{adj}}$ is the adjacent matrix of the input graph. Degree matrix $\mathbf{D}$ is a diagonal matrix with diagonal element $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}^{\text{adj}}$. $\mathbf{U}$ is the matrix of eigenvectors and $\Lambda = \text{diag}[\lambda 1, \ldots, \lambda n]$ is the diagonal matrix of eigenvalues. However, it is computation-intensive to perform eigenvalue decomposition on a large-scale Laplacian matrix. As a result, we use Chebyshev polynomial approximation [34] to reduce computation

$$\mathbf{U}g_\theta(\Lambda)\mathbf{U}^T\mathbf{F}^{l} \approx \sum_{k=0}^{K-1}\theta_k T_k\left(\tilde{\mathbf{L}}\right)\mathbf{F}^{l} \tag{9}$$

where $\theta \in \mathbb{R}^K$ is the vector of polynomial coefficients to be updated during model training. $T_k(\tilde{\mathbf{L}}) \in \mathbb{R}^{n\times n}$ is the Chebyshev polynomial of order k evaluated at $\tilde{\mathbf{L}} = (2/\lambda_{\max})\mathbf{L} - \mathbf{I}_N$. $T_k(\tilde{\mathbf{L}})$ can be computed recursively using the $T_0(\tilde{\mathbf{L}})$ and $T_1(\tilde{\mathbf{L}})$. $\lambda_{\max}$ is the largest eigenvalue of the Laplacian matrix. K is the largest order of used in the approximation.

The spatial attention matrix $\mathbf{S}'$ is used to adjust elements in $T_k(\tilde{\mathbf{L}})$ and the graph convolution operation becomes

$$\mathbf{F}^{\text{inter}} = \sum_{k=0}^{K-1}\theta_k T_k\left(\tilde{\mathbf{L}}\right) \odot \mathbf{S}'\mathbf{F}^{O1}. \tag{10}$$

By choosing $K < n$ in the Chebyshev polynomial approximation, we reduce the number of trainable parameters and accelerate the training efficiency.

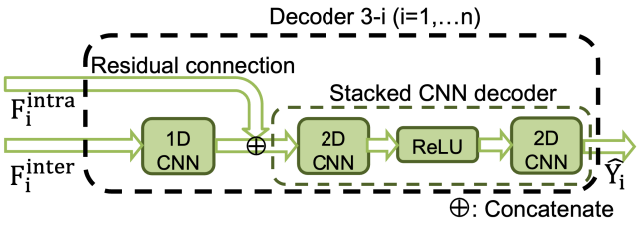


Fig. 5. Architecture of one decoder in stage 3. Stage 3 merges intracluster and intercluster features in the previous two stages and decodes them with stacked CNNs to make the final prediction.

D. Stage 3: Decoder

The output of stage 2 is a graph sequence where each vertex represents features extracted from the corresponding sensor cluster. As our objective is to recover missing data for each sensor, it is necessary to decompose the features of each vertex in stage 2 back into features for multiple sensors within the sensor cluster. Fig. 5 illustrates the architecture of one such decoder. Mirroring the CME in stage 1, which merges features of multiple sensors, we employ a 1-D CNN in stage 3 to decompose the output vertex features of stage 2.

Given the potential loss of local information during the merging and decomposing processes, we incorporate a residual connection from stages 1–3. Specifically, the intracluster vertex features $\mathbf{F}_i^{\text{intra}}$ in stage 1 are concatenated with the output of the 1-D CNN in stage 3. The concatenated features are then processed by the stacked CNN decoder. The output of the decoder for the ith sensor cluster is denoted as $\hat{\mathbf{Y}}_i$. Finally, the imputed data for each sensor cluster are computed according to (2).

In the MSA-GCN model, we leverage predictions from both stages 1 and 3 from n sensor clusters to compute the overall loss. The final loss function is computed as $\mathbf{Loss} = \sum_{i=1}^{n} \mathbf{Loss}_i$, where

$$\begin{aligned}\mathbf{Loss}_i = {} & \mathbf{mse}\Big(\mathbf{F}_i^{\text{input}}, \mathbf{Impute}\Big(\mathbf{F}_i^{\text{input}}, \hat{\mathbf{Y}}_i, \mathbf{M}_i\Big)\Big) \\ & + \mathbf{mse}\Big(\mathbf{F}_i^{\text{input}}, \mathbf{Impute}\Big(\mathbf{F}_i^{\text{input}}, \hat{\mathbf{Y}}_i^{\text{pred1,fwd}}, \mathbf{M}_i\Big)\Big) \\ & + \mathbf{mse}\Big(\mathbf{F}_i^{\text{input}}, \mathbf{Impute}\Big(\mathbf{F}_i^{\text{input}}, \hat{\mathbf{Y}}_i^{\text{pred1,bwd}}, \mathbf{M}_i\Big)\Big) \\ & + \mathbf{mse}\Big(\mathbf{F}_i^{\text{input}}, \mathbf{Impute}\Big(\mathbf{F}_i^{\text{input}}, \hat{\mathbf{Y}}_i^{\text{pred2,fwd}}, \mathbf{M}_i\Big)\Big) \\ & + \mathbf{mse}\Big(\mathbf{F}_i^{\text{input}}, \mathbf{Impute}\Big(\mathbf{F}_i^{\text{input}}, \hat{\mathbf{Y}}_i^{\text{pred2,bwd}}, \mathbf{M}_i\Big)\Big)\end{aligned} \tag{11}$$

and **mse** denotes mean square error loss function.

V. Experiments With SHM Data and Weather Data

A. Graph Representation of Data

We utilized historical monitoring data from the SHM system of Tsing Ma Bridge, a large-span suspension bridge in Hong Kong, to evaluate the data imputation performance of the MSA-GCN. The SHM system continuously collects dynamic responses from multiple substructures of the bridge, including the bridge deck, cables, and abutments. In addition, it gathers ambient environmental conditions around the bridge, such as wind velocity and temperature. Accelerometers mounted on the bridge deck also record acceleration in two directions: vertical and cross-deck. After discarding the abnormal data from the raw dataset, we curated a dataset containing six monitoring items, as detailed in Table I.

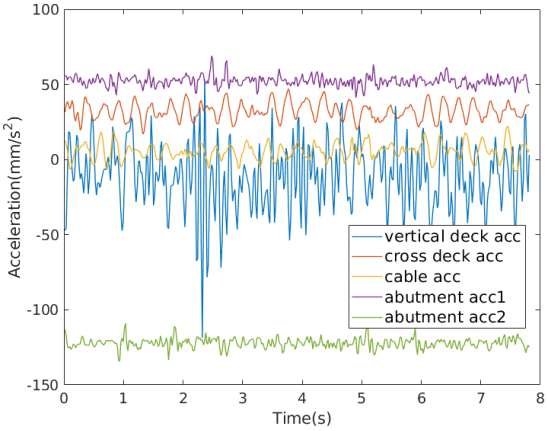


Fig. 6. Comparison of data in different monitoring items. Acceleration of cable, deck, and abutment exhibit different vibration patterns over time. However, acceleration collected by two sensors on the same abutment pattern has similar vibration patterns.

TABLE I
Data Characteristics in Each Subsystem Monitoring Item

Monigtoring item	Number of sensor	Sampling frequency	Unit
Abutment acceleration	3	51.2 Hz	mm/s^2
Cable acceleration	6	51.2 Hz	mm/s^2
Cross deck direction acceleration	5	51.2 Hz	mm/s^2
Vertical deck direction acceleration	10	51.2 Hz	mm/s^2
Wind velocity	11	2.56 Hz	m/s
Temperature	103	0.07 Hz	Degs C

Fig. 6 illustrates the different vibration patterns of the bridge's cables, deck, and abutments. The figure aligns with our hypothesis that data from the same monitoring item usually exhibit similar characteristics, while data from different items vary. The two acceleration time series from the bridge abutment show a similar vibration pattern. However, the vibrations of the abutment and deck in the vertical direction are more frequent than those of the cables and deck in the cross-deck direction. The acceleration in the vertical deck direction also has a larger amplitude than the other three items. This could be explained by the fact that a bridge's vertical stiffness is typically smaller than its stiffness in other directions due to the bridge's long span. It should be noted that offsets exist in each accelerator on the bridge, so the mean value of the measured accelerations is not zero. However, the acceleration responses used during model training have a zero mean value because we normalize them before training. As a result, the offset of accelerators will not impact the performance of our proposed method.

We create six graphs using the time series from the six monitoring items. Specifically, we design graphs for the four accelerometer clusters based on the sensor layout on the bridge. Edges are assigned to sensor pairs whose distance is less than a predefined threshold. For wind velocity and temperature data, we follow the same procedure of computing the adjacent matrix in [27]. We compute the Euclidean

TABLE II
DATA IMPUTATION ERROR OF MSA-GCN AND BASELINE MODELS IN SHM DATA

Method	Self-designed graphs in stage 1						Fully connected graphs in stage 1					
	$p_{block} = 0.01$		$p_{block} = 0.02$		$p_{block} = 0.03$		$p_{block} = 0.01$		$p_{block} = 0.02$		$p_{block} = 0.03$	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
Global Mean	13.376	555.137	13.373	554.831	13.370	554.627	/	/	/	/	/	/
Batch Mean	9.951	332.754	9.951	332.863	9.949	332.649	/	/	/	/	/	/
Interpolation	7.340	231.789	8.236	281.992	8.671	305.701	/	/	/	/	/	/
BRITS	8.251	240.748	9.011	279.498	9.379	297.293	/	/	/	/	/	/
GRIN	4.863	106.546	5.846	136.660	6.357	153.040	4.523	99.715	5.556	129.120	6.081	144.556
Transformer	7.533	201.789	8.146	233.688	8.483	250.388	/	/	/	/	/	/
TimesNet	5.572	108.164	5.964	123.294	6.199	132.627	/	/	/	/	/	/
MS-GCN	4.352	74.662	5.056	95.218	5.441	107.218	4.633	84.890	5.378	107.778	5.781	121.098
MSA-GCN w/o stage 1	13.289	551.958	13.295	552.849	13.289	552.513	13.291	543.362	13.296	544.213	13.291	543.883
MSA-GCN w/o stage 2	4.327	71.384	4.994	90.468	5.355	101.443	4.727	87.639	5.474	111.122	5.881	124.637
MSA-GCN	4.248	67.963	4.906	86.438	5.265	97.135	4.627	81.915	5.366	104.698	5.766	118.083

Note: Transformer denotes Non-stationary Transformer

distance between sensor pairs to create a similarity matrix. The connectivity threshold is set at the 75th percentile of each row in the similarity matrix, ensuring that only the strongest connections are retained. Furthermore, we apply second-level filtering by setting the overall 75th percentile of the similarity matrix as a cutoff, discarding values below this threshold. In each graph, the attributes of each vertex are the time-series data collected from each sensor. We then divide these time series into segments with a fixed length of 28.125 s for this experiment. Given that the sampling frequency for all the accelerometers is 51.2 Hz, there are 1440 time steps in each acceleration segment. The length of each wind velocity and temperature data segment is 256 and 2 steps, respectively. Finally, the generated dataset comprises 46 080 samples, representing 15 days of SHM monitoring data.

B. Baseline Models

We consider seven baseline models as alternatives to MSA-GCN for comparison, including global mean value, batch mean value, linear interpolation, BRITS [25], GRIN [32], nonstationary transformer [35], and TimesNet [36]. The global mean value and batch mean value are naive data imputation methods that use the mean value of the training set and each batch, respectively. Linear interpolation is a method that constructs a straight line passing through two known data points and uses that line to interpolate the missing value. BRITS and GRIN are two deep learning-based models known for their state-of-the-art data imputation accuracy. BRITS is based on a RNN, while GRIN is based on a message-passing GRU. GRIN's performance has been validated on several public datasets for MTS imputation.

We design three variants based on MSA-GCN as ablation experiments to validate the contribution of multilayer architecture and spatial attention mechanism. Specifically, we replace the ASGCN in stage 2 with a traditional GCN. We call this model multistage GCN (MS-GCN). In MS-GCN, we formulate a fully connected graph in stage 2 with the output of stage 1. The GCN in stage 2 has three graph convolution layers with a hidden size of 1440, 256, and 1440. Moreover, we remove stages 1 and 2 from MSA-GCN, respectively, and the models are named MSA-GCN w/o stage 1 and MSA-GCN w/o stage 2. It should be noted here that we still keep DME and CME in stage 1 in MSA-GCN w/o stage 1. Because they are used to transform data from different sensor clusters into consistent shapes. Then, the heterogeneous data can be fed into the intercluster encoder in stage 2.

C. Dataset Generation and Model Training

To ensure stable training, we normalize each time series data to have zero mean and unit variance before they are input to MSA-GNN. As mentioned in Section III-A, we manually remove data to simulate data loss. In the case of "block missing," p_{block} is set to 0.02 during model training. The minimum and maximum durations for block failure are 3 and 10 steps, respectively. During testing, we use different p_{block} values to test the generalization ability of our method under varying severities of data loss. We set the batch size to 32 and the initial learning rate to 0.001. During training, we optimize the learning rate using the Adam Optimizer with a Cosine Annealing Scheduler.

Both MSA-GCN and the baseline models are trained with the same dataset and model training hyperparameters. p_{block} is set to 0.02 in the training set. It's worth noting that we use three models independently to process acceleration, wind velocity, and temperature data in our training set while using BRITS, GRIN, nonstationary transformer, and TimesNet. Because the acceleration, wind velocity, and temperature data in the training set consist of matrices with three different shapes, corresponding to the three different sampling frequencies in Table I. The three trained models are used to recover data for the corresponding sensor clusters, respectively, with their imputation error calculated from all sensor clusters. The overall experimental results are summarized in Table II.

We implement MSA-GCN using PyTorch 1.12.1. Graph operations are performed using DGL 0.9.1, an open-source library for deep learning on graphs. All models are trained using a GTX 3090Ti GPU.

D. Data Imputation Performance Comparison

We evaluate the data imputation accuracy of the various methods using mean absolute error (MAE) and mean squared

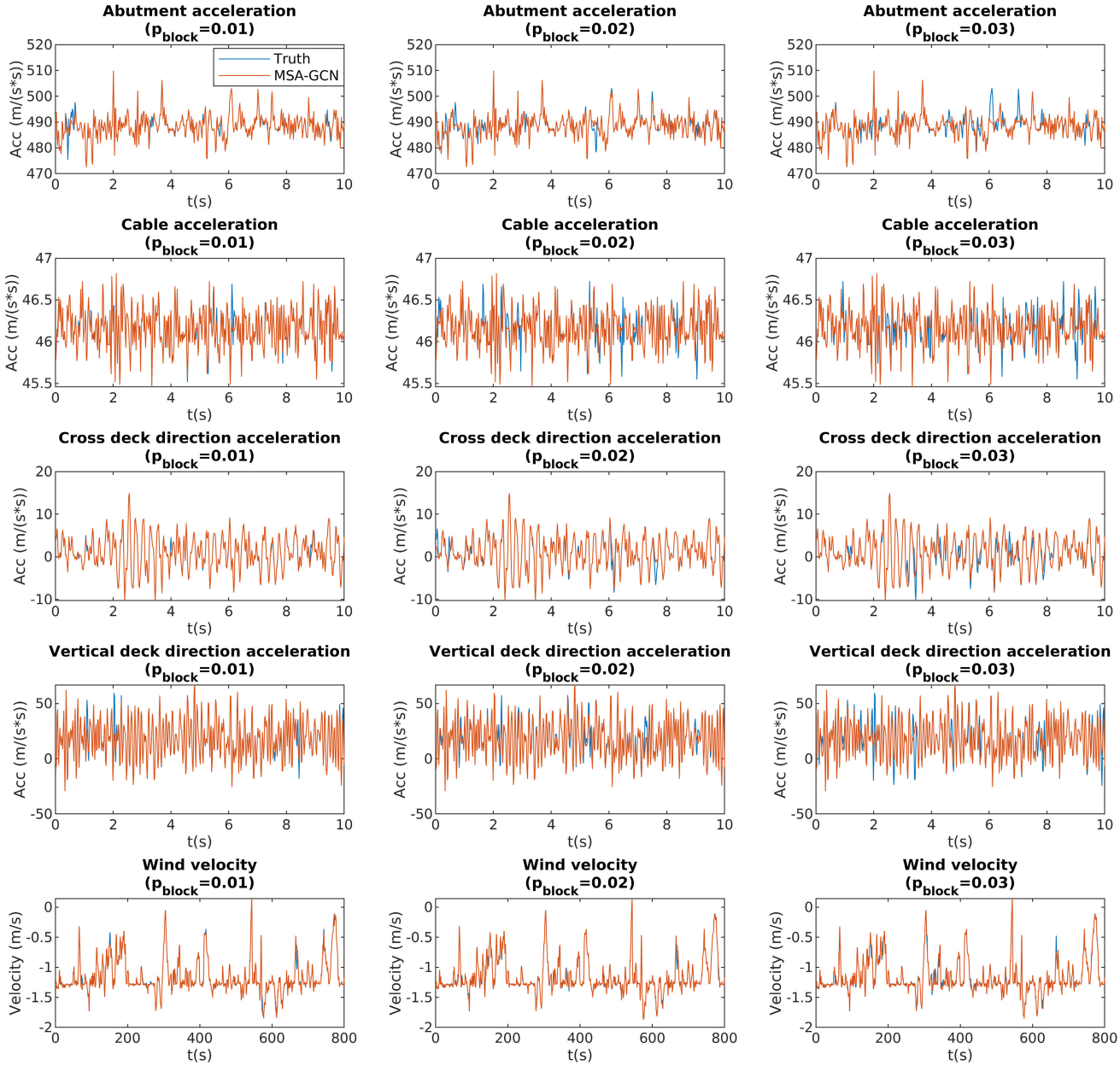


Fig. 7. Comparison of data in the time domain. With the increase of p_{block}, the data imputation error of each monitoring item also increases.

TABLE III
DATA IMPUTATION ERROR OF MSA-GCN AND BASELINE MODELS IN WEATHER DATA

Method	Self-designed graphs in stage 1						Fully connected graphs in stage 1					
	$p_{block}=0.01$		$p_{block}=0.02$		$p_{block}=0.03$		$p_{block}=0.01$		$p_{block}=0.02$		$p_{block}=0.03$	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
Global Mean	37.554	7825.589	40.107	8607.056	37.381	7778.770	/	/	/	/	/	/
Batch Mean	17.332	3395.457	17.601	3459.153	16.566	3172.549	/	/	/	/	/	/
Interpolation	3.938	876.138	3.842	940.960	3.836	716.601	/	/	/	/	/	/
Transformer	8.611	673.208	8.521	677.738	8.644	673.208	/	/	/	/	/	/
TimesNet	5.859	553.320	6.081	599.962	6.361	654.788	/	/	/	/	/	/
BRITS	6.097	515.401	6.467	575.999	6.269	493.790	/	/	/	/	/	/
GRIN	5.276	514.239	5.518	493.976	5.622	448.377	5.528	630.101	5.782	632.487	6.000	574.550
MSA-GCN	5.182	477.953	5.558	528.255	5.777	560.203	/	/	/	/	/	/

Note: Transformer denotes Non-stationary Transformer

error (mse). The prediction accuracy of these methods is assessed using the same testing sets. To explore how the structure of the input graph-shaped data influences the final imputation accuracy, we tested both self-designed graphs and

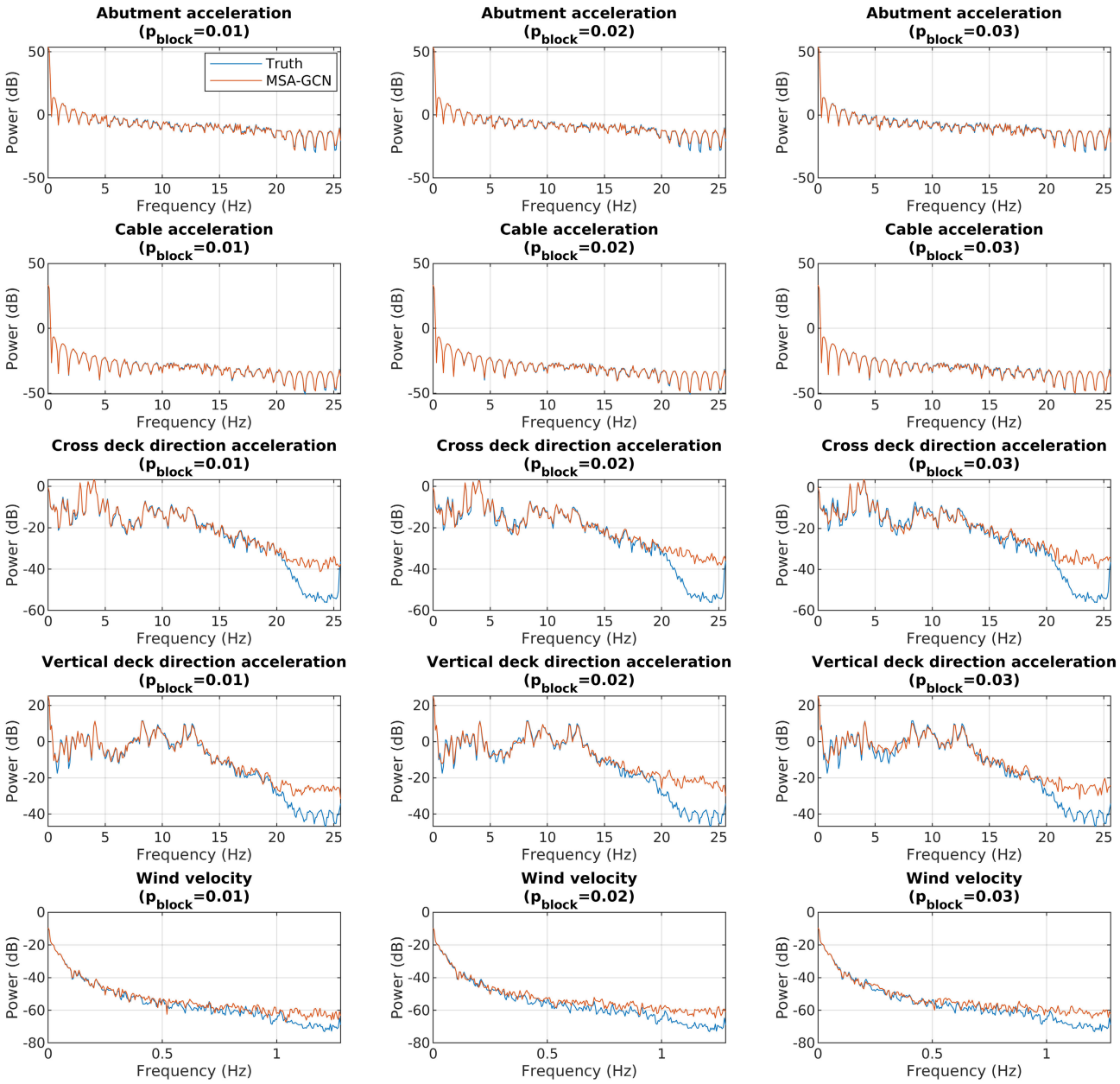


Fig. 8. Comparison of data in the frequency domain. With the increase of p_{block}, the difference of spectrum between MSA-GCN recovered data and ground truth also increases. Their peaks in the spectrum match well, demonstrating that frequency features of data are well preserved in MSA-GCN's prediction.

fully connected graphs for all the GNN-based models. As mentioned in Section V-A, we design a self-designed graph based on the layout of sensors in the bridge. In a fully connected graph, each vertex is connected with all other vertices. To test the generalization of the models, we generate three testing sets with different p_{block} values: 0.01, 0.02, and 0.03. These three testing sets represent the varying severity of data loss in real-world data.

As shown in Table II, MSA-GCN outperforms all other models across different scenarios, achieving the smallest MAE and mse. Compared to GRIN, which yielded the smallest data imputation error among the five baseline models, MSA-GCN reduces MAE by up to 20.75% and mse by up to 58.10%. Furthermore, the prediction error of MSA-GCN remains stable across different p_{block} values, demonstrating its robustness under varying data loss severities.

Despite GRIN's impressive accuracy in public datasets with homogeneous data, its imputation performance in heterogeneous SHM data is less satisfactory. Using fully connected graphs allows GRIN to reduce the error, but it still yields larger data imputation errors than MSA-GCN and MS-GCN. A likely explanation is that GRIN does not incorporate a multistage architecture and thus cannot fully exploit the heterogeneous and dynamic correlations in the data. The prediction results of Global Mean, Batch Mean, and Interpolation with fully connected graphs are blank because these methods do not require transforming input data into graphs. Global Mean yields the largest imputation error, demonstrating the poor

performance of naive imputation methods on heterogeneous data.

Both MSA-GCN and MS-GCN perform better with self-designed graphs than with fully connected graphs in terms of imputation accuracy. The reason can be that the multistage architecture already efficiently captures the heterogeneous correlations in the input data. Fully connected graphs link uncorrelated nodes, leading to overfitting of the two multistage models.

MS-GCN yields higher prediction errors than MSA-GCN across various data loss severities, demonstrating the value of the spatial attention mechanism employed in ASGCN of stage 2. MSA-GCN w/o stages 1 and 2 also demonstrate larger errors than the complete MSA-GCN, reinforcing the importance of intracluster and intercluster features in predicting missing values. Despite their subpar performance relative to MSA-GCN, both MS-GCN and MSA-GCN w/o stage 2 outperform other baseline models, affirming the superiority of the multistage architecture and intracluster features in restoring missing values.

E. Data Imputation Results Discussion

Fig. 7 displays the ground truth data and the data imputed by MSA-GCN for the five monitoring items under different data loss severities. MSA-GCN achieves impressive accuracy in each monitoring item, with some exceptions for local peaks in the time series. As p_{block} increases, the imputation error also increases. However, the error remains within acceptable limits even when p_{block} increases to 0.03, where almost 18% of the raw data are missing.

We also compare the ground truth and recovered data in the frequency domain. We use fast Fourier transform to estimate the power spectral density (PSD) of four types of acceleration data. As shown in Fig. 8, the spectrum of the ground truth and the MSA-GCN recovered data match very well. As p_{block} increases, the PSD of the MSA-GCN imputed data gradually deviates from the ground truth. However, the peaks in the PSD are preserved accurately. This shows the great potential of MSA-GCN in real-world implementation for SHM systems. Because peaks in the PSD of acceleration represent the vibration frequencies of an engineering structure. These are crucial features used for estimating the health condition of engineering structures.

Fig. 7 indicates that MSA-GCN incurs a higher imputation error in consecutive data loss steps than in a single step. To analyze how the length of the data loss step influences MSA-GCN's performance, we compute its imputation error under different lengths of consecutive data loss time steps. Fig. 9 shows that both the MAE and mse of error increase with the increase of consecutive data loss steps for each monitoring item.

Moreover, within each time step, the number of failed sensors can also influence the data imputation accuracy. This is because the more sensors fail at the same time, the fewer spatial correlations can be used to recover the missing value. For instance, if we have ten sensors to monitor acceleration in the vertical deck direction, and data from six sensors are missing in one time step, we have to recover them with the remaining 4 sensors. This will be more challenging than if only two sensors are missing. We computed and summarized the MAE and mse of error for each time step. Fig. 10 shows the change of error with respect to the increase of failed sensors in a one-time step. For variables such as abutment acceleration, cable acceleration, acceleration in the cross-deck direction, acceleration in the vertical deck direction, and temperature, an increased number of failed sensors consistently results in larger imputation errors. However, the situation differs in the case of wind velocity data where the number of failed sensors does not seem to significantly impact the data imputation error. This can be attributed to the weaker spatial correlations within multipoint wind velocity data, while their temporal correlations are strong. As a result, even if most of the other sensors fail, MSA-GCN can effectively recover a sensor's missing wind velocity data using its past or future measurements.

Finally, we would like to analyze how the error performance with the continuous increase of data loss severity. We increase p_{block} to 0.1, in which around 36% of raw data are missing. Fig. 11 shows that the overall imputation error continues to increase with respect to the increase of p_{block}. It also shows that the increase of imputation error gradually becomes stable when p_{block} becomes larger.

F. Data Imputation Experiments on Weather Dataset

We also test the data imputation performance of our proposed MSA-GCN in a public multivariate dataset with baseline models. We chose the Wetterstation weather dataset [37] because this dataset is heterogeneous. Different variates come from different monitoring items, including air temperature, humidity, and air pressure. However, each monitoring item in this dataset has only one sensor and all monitoring items share the same sampling frequency, making this dataset less heterogeneous than our SHM dataset used in Section V-A.

We test the performance of all baseline models used in experiments with the SHM dataset. Their performances are summarized in Table III. The table demonstrates that MSA-GCN surpasses other baseline models in imputation accuracy on the Wetterstation weather dataset. However, the performance between MSA-GCN and GRIN is closely matched. The distinct advantage of MSA-GCN in handling heterogeneous datasets is not fully showcased in this dataset due to uniform sampling frequencies across sensors, rendering it less heterogeneous compared to SHM datasets where MSA-GCN typically excels. Besides, the results for MSA-GCN in the column of fully connected graphs in stage 1 are blank because each monitoring item in the Wetterstation weather dataset only has one sensor. There is only one node in each graph in stage 1, making self-designed and fully connected graphs the same.

The data imputation MAE and mse computed in our experiment are higher than those in experiments of TimesNet [36]. That arises from differences in experimental settings and data normalization. In our experiments, we calculate MAE and mse for un-normalized data, whereas normalized data are used to compute MAE and mse in experiments of TimesNet. In addition, our experiments account for both block-missing

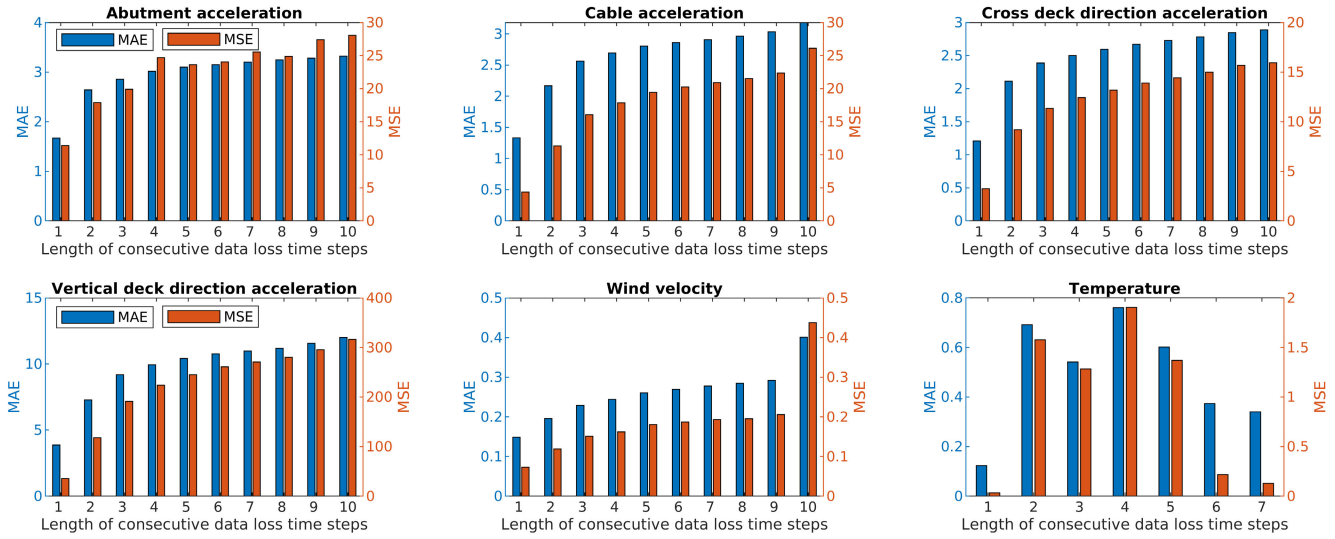


Fig. 9. Data imputation error with different lengths of data loss steps. In most monitoring items, the error increases with the increase of data loss steps.

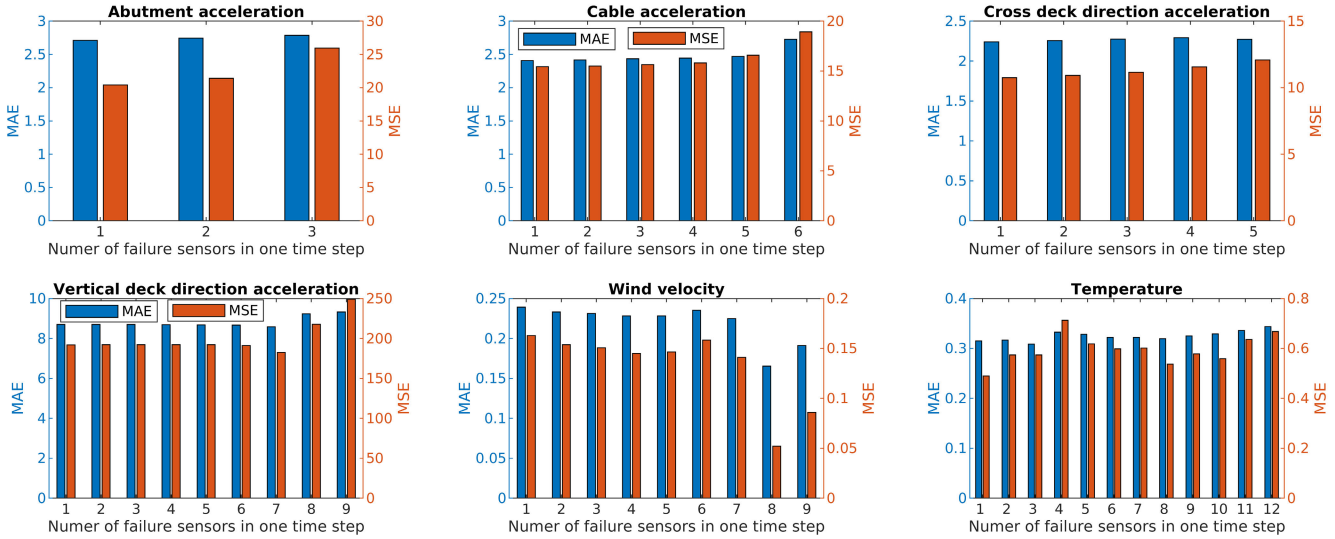


Fig. 10. Data imputation error with the different number of failure sensors. In most monitoring items, the error increases with the increase of failure sensors.

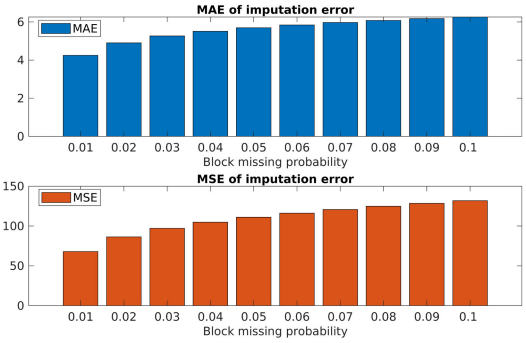


Fig. 11. Data imputation error under increasing p_{block}. The error gets larger with the increase of p_{block} and gradually converges to a constant value.

and point-missing scenarios across various variates, adding complexity to our data imputation tasks. In contrast, experiments in TimesNet only include point-missing scenarios. Block-missing scenarios, where data is lost consecutively, pose a greater challenge for accurate data recovery than isolated point-missing scenarios.

VI. Concluding Remarks

In this article, we introduced MSA-GCN, a novel approach for imputing missing data in MTS. Unlike many existing methods that focus on homogeneous data, MSA-GCN is designed to handle heterogeneous data from multiple monitoring items, making it more applicable to real-world data loss situations. By incorporating a multistage architecture, MSA-GCN is capable of learning heterogeneous correlations between MTS. It captures diverse intracluster correlations with multiple encoders in stage 1 and intercluster correlations in stage 2. Furthermore, MSA-GCN integrates a spatial attention layer in stage 2, enabling the model to capture dynamic intercluster correlations effectively.

Our experimental results, using six types of monitoring data from the Tsing Ma Bridge, demonstrate MSA-GCN's superior

data imputation accuracy compared to baseline models under various data loss severities. Ablation experiments also confirm the effectiveness of the spatial attention mechanism in MSA-GCN. The frequency domain features of the imputed data closely align with the ground truth, emphasizing the practical value of MSA-GCN in real-world SHM systems. Another experiment on the Wetterstation weather dataset further validates MSA-GCN's superiority over baseline methods in terms of imputation accuracy. As a future direction, we plan to extend MSA-GCN to handle even more heterogeneous monitoring items, building upon the foundation established in this work. In addition, we will investigate the applicability of MSA-GCN to other domains, such as healthcare, finance, and environmental monitoring, where heterogeneous MTS data are also prevalent.

ACKNOWLEDGMENT

The authors wish to thank Hong Kong SAR Government Highways Department for providing the long-term SHM data for the Tsing Ma Bridge. The findings and opinions expressed in this article are from them alone and are not necessarily the views of the sponsors.

REFERENCES

[1] Z. Chen, H. Li, and Y. Bao, "Analyzing and modeling inter-sensor relationships for strain monitoring data and missing data imputation: A copula and functional data-analytic approach," *Struct. Health Monitor.*, vol. 18, no. 4, pp. 1168–1188, Jul. 2019.

[2] W. Lu, J. Teng, C. Li, and Y. Cui, "Reconstruction to sensor measurements based on a correlation model of monitoring data," *Appl. Sci.*, vol. 7, no. 3, p. 243, Mar. 2017.

[3] Q. Shang, Z. Yang, S. Gao, and D. Tan, "An imputation method for missing traffic data based on FCM optimized by PSO-SVR," *J. Adv. Transp.*, vol. 2018, no. 1, pp. 1–21, 2018.

[4] X. Chen, S. Liang, Z. Zhang, and F. Zhao, "A novel spatiotemporal data low-rank imputation approach for traffic sensor network," *IEEE Internet Things J.*, vol. 9, no. 20, pp. 20122–20135, Oct. 2022.

[5] Y. Liu, T. Dillon, W. Yu, W. Rahayu, and F. Mostafa, "Missing value imputation for industrial IoT sensor data with large gaps," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 6855–6867, Aug. 2020.

[6] Y. Yu, V. O. K. Li, and J. C. K. Lam, "Hierarchical recovery of missing air pollution data via improved long-short term context encoder network," *IEEE Trans. Big Data*, vol. 9, no. 1, pp. 93–105, Feb. 2023.

[7] Z. Wu, C. Ma, X. Shi, L. Wu, Y. Dong, and M. Stojmenovic, "Imputing missing indoor air quality data with inverse mapping generative adversarial network," *Building Environ.*, vol. 215, May 2022, Art. no. 108896.

[8] Z. Karevan, "Data-driven prediction of complex systems," Ph.D. thesis, Fac. Eng. Sci., KU Leuven, Belgium, 2018. [Online]. Available: https://lirias.kuleuven.be/retrieve/523765

[9] X.-Y. Huang et al., "Multi-matrices factorization with application to missing sensor data imputation," *Sensors*, vol. 13, no. 11, pp. 15172–15186, Nov. 2013.

[10] E. Afrifa-Yamoah, U. A. Mueller, S. M. Taylor, and A. J. Fisher, "Missing data imputation of high-resolution temporal climate time series data," *Meteorolog. Appl.*, vol. 27, no. 1, Jan. 2020, Art. no. e1873.

[11] D. Li, J. Deogun, W. Spaulding, and B. Shuart, "Towards missing data imputation: A study of fuzzy K-means clustering method," in *Proc. Int. Conf. Rough Sets Current Trends Comput.*, Uppsala, Sweden. Cham, Switzerland: Springer, 2004, pp. 573–579.

[12] T. Emmanuel, T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, and O. Tabona, "A survey on missing data in machine learning," *J. Big Data*, vol. 8, no. 1, pp. 1–37, Oct. 2021.

[13] G. Fan, J. Li, and H. Hao, "Lost data recovery for structural health monitoring based on convolutional neural networks," *Struct. Control Health Monitor.*, vol. 26, no. 10, Oct. 2019, Art. no. e2433.

[14] G. Fan, J. Li, and H. Hao, "Dynamic response reconstruction for structural health monitoring using densely connected convolutional networks," *Struct. Health Monitor.*, vol. 20, no. 4, pp. 1373–1391, Jul. 2021.

[15] Z. Tang, Y. Bao, and H. Li, "Group sparsity-aware convolutional neural network for continuous missing data recovery of structural health monitoring," *Struct. Health Monitor.*, vol. 20, no. 4, pp. 1738–1759, Jul. 2021.

[16] X. Lei, L. Sun, and Y. Xia, "Lost data reconstruction for structural health monitoring using deep convolutional generative adversarial networks," *Struct. Health Monitor.*, vol. 20, no. 4, pp. 2069–2087, Jul. 2021.

[17] Y. Bao, Z. Tang, and H. Li, "Compressive-sensing data reconstruction for structural health monitoring: A machine-learning approach," *Struct. Health Monitor.*, vol. 19, no. 1, pp. 293–304, Jan. 2020.

[18] H. Jiang, C. Wan, K. Yang, Y. Ding, and S. Xue, "Continuous missing data imputation with incomplete dataset by generative adversarial networks–based unsupervised learning for long-term bridge health monitoring," *Struct. Health Monitor.*, vol. 21, no. 3, pp. 1093–1109, May 2022.

[19] H.-P. Wan and Y.-Q. Ni, "Bayesian multi-task learning methodology for reconstruction of structural health monitoring data," *Struct. Health Monitor.*, vol. 18, no. 4, pp. 1282–1309, Jul. 2019.

[20] D. Cao et al., "Spectral temporal graph neural network for multivariate time-series forecasting," in *Proc. NIPS*, 2020, pp. 17766–17778.

[21] J. Niu, S. Li, and Z. Li, "Restoration of missing structural health monitoring data using spatiotemporal graph attention networks," *Struct. Health Monitor.*, vol. 21, no. 5, pp. 2408–2419, Sep. 2022.

[22] X. Yi, Y. Zheng, J. Zhang, and T. Li, "ST-MVL: Filling missing values in geo-sensory time series data," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 2704–2710.

[23] Y. Chen, Z. Li, C. Yang, X. Wang, G. Long, and G. Xu, "Adaptive graph recurrent network for multivariate time series imputation," in *Proc. Int. Conf. Neural Inf. Process.*, 2022, pp. 64–73.

[24] W. Liang et al., "Spatial–temporal aware inductive graph neural network for C-ITS data recovery," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 8, pp. 8431–8442, 2022.

[25] W. Cao, D. Wang, J. Li, H. Zhou, L. Li, and Y. Li, "BRITS: Bidirectional recurrent imputation for time series," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.

[26] Y.-F. Zhang, P. J. Thorburn, W. Xiang, and P. Fitch, "SSIM—A deep learning approach for recovering missing time series sensor data," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 6618–6628, Aug. 2019.

[27] I. Spinelli, S. Scardapane, and A. Uncini, "Missing data imputation with adversarially-trained graph convolutional networks," *Neural Netw.*, vol. 129, pp. 249–260, Sep. 2020.

[28] J. You, X. Ma, Y. Ding, M. J. Kochenderfer, and J. Leskovec, "Handling missing data with graph representation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 19075–19087.

[29] G. Fan, J. Li, H. Hao, and Y. Xin, "Data driven structural dynamic response reconstruction using segment based generative adversarial networks," *Eng. Struct.*, vol. 234, May 2021, Art. no. 111970.

[30] Y. Luo et al., "Multivariate time series imputation with generative adversarial networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.

[31] J. Hou et al., "Deep learning and data augmentation based data imputation for structural health monitoring system in multi-sensor damaged state," *Measurement*, vol. 196, Jun. 2022, Art. no. 111206.

[32] A. Cini, I. Marisca, and C. Alippi, "Filling the g_ap_s: Multivariate time series imputation by graph neural networks," 2021, *arXiv:2108.00298*.

[33] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial–temporal graph convolutional networks for traffic flow forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, no. 1, pp. 922–929.

[34] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016.

[35] Y. Liu, H. Wu, J. Wang, and M. Long, "Non-stationary transformers: Exploring the stationarity in time series forecasting," in *Proc. 36th Conf. Neural Inf. Process. Syst.*, 2022, pp. 9881–9893.

[36] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "TimesNet: Temporal 2D-variation modeling for general time series analysis," in *Proc. 11th Int. Conf. Learn. Represent.*, 2022.

[37] Wetterstation. *Weather Dataset*. Accessed: Apr. 22, 2024. [Online]. Available: https://www.bgc-jena.mpg.de/wetter/weather_data.html

Qianyi Chen received the B.S. degree from Huazhong University of Science and Technology, Wuhan, China, in 2016, the M.S. degree from Zhejiang University, Hangzhou, China, in 2019, and the Ph.D. degree from The Hong Kong Polytechnic University, Hong Kong, China, in 2024.

He is currently a Post-Doctoral Researcher with the School of Engineering, Westlake University, Hangzhou. His research interests include AI for science, edge computing, and structural health monitoring.

Jiannong Cao (Fellow, IEEE) received the M.Sc. and Ph.D. degrees in computer science from Washington State University, Pullman, WA, USA, in 1986 and 1990, respectively.

He is currently the Otto Poon Charitable Foundation Professor in data science and the Chair Professor with the Department of Computing, The Hong Kong Polytechnic University (PolyU), Hong Kong, where he is also the Dean of the Graduate School, the Director of the Research Institute of Artificial Intelligent of Things, and the Vice Director of the University's Research Facility in Big Data Analytics. His current research interests include wireless sensing and networking, big data and machine learning, and mobile cloud and edge computing.

Dr. Cao is a member of the Academia Europaea and an ACM Distinguished Member. He served as the Chair and member of organizing and technical committees for many international conferences, such as IEEE INFOCOM and IEEE PERCOM, and as an Associate Editor for many international journals, such as IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, and IEEE TRANSACTIONS ON BIG DATA.

Yu Yang received the B.Eng. degree in computer science from Xi'an University of Science and Technology, Xi'an, China, in 2012, the M.Eng. degree in pattern recognition and intelligence systems from Shenzhen University, Shenzhen, China, in 2015, and the Ph.D. degree in computer science from The Hong Kong Polytechnic University, Hong Kong, in 2021.

He is currently an Assistant Professor with the Centre for Learning, Teaching and Technology (LTTC), The Education University of Hong Kong (EdUHK), Hong Kong. His research interests include graph representation learning, spatiotemporal data analysis, urban computing, and learning analytics.

Dr. Yang served as a reviewer and Program Committee Member for many international journals and conferences like IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, *ACM Transactions on Information Systems*, SIGKDD, TheWebConf, AAAI, NeurIPS, and ICML.

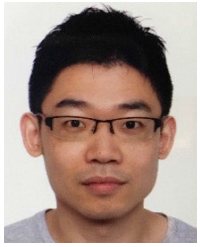
Wanyu Lin (Member, IEEE) received the B.Engr. degree from the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan, China, in 2012, the M.Phil. degree from the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, in 2015, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada, in 2020.

She is an Assistant Professor with the Department of Data Science and Artificial Intelligence and the Department of Computing, The Hong Kong Polytechnic University. Her research interests include AI for science and explainable AI.

Dr. Lin is serving as an Associate Editor for IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS (TNNLS).

Sumei Wang received the B.Eng. degree from Hebei University, Baoding, China, in 2013, and the Ph.D. degree from Zhejiang University, Hangzhou, China, in 2018.

She is currently a Research Assistant Professor with The Hong Kong Polytechnic University, Hong Kong. Her research interests include structural dynamics, rail-bridge interaction, machine vision, and structural health monitoring in railway engineering.



Youwu Wang received the B.Eng. degree from Tianjin University of Science and Technology, Tianjin, China, in 2008, the M.Eng. degree from Hunan University, Changsha, China, in 2012, and the Ph.D. degree from PolyU in 2017.

He is currently a Research Assistant Professor with the Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University (PolyU), Hong Kong, China. His research interests include structural health monitoring, Bayesian inference, and structural dynamics.