Received: February 17, 2024. Revised: March 15, 2024. 352

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Spatio-Temporal Bi-LSTM based Variational Auto-Encoder for Multivariate IoT Data Imputation

Venkata Vidyalakshmi Guggilam[1]* **Gopikrishnan Sundaram[1]**

[1]*School of Computer Science and Engineering, Vellore Institute of Technology-Andhra Pradesh University, Guntur, Andhra Pradesh, India*
* Corresponding author's Email: vidyalakshmi.21phd7089@vitap.ac.in

Abstract: In the relam of the Internet of Things (IoT), prevalence of missing data due to continuous data collection by smart devices necessitates the essential preliminary step of data imputation before engaging in information mining activities. IoT data exhibit robust interconnections in both spatial and temporal dimensions, surpassing the limitations of Euclidean space. Yet, prevailing machine learning and deep learning approaches often focus solely on temporal attributes or capture spatial features exclusively within a Euclidean framework. To address these challenges, this paper introduces a novel network named ST-Bi-LSTM-VAE (Spatio-Temporal Bidirectional Long Short-Term Memory based Variational Auto-Encoder). The architecture of ST-Bi-LSTM-VAE is primarily grounded in the Variational Auto-Encoder (VAE) framework. This innovative approach incorporates two distinct types of VAEs. The first type is dedicated to computing the adjacent matrix of the device network, a crucial input for the Graph Convolutional Network (GCN) essential in capturing intricate spatial relationships among devices. The second type of VAE is specifically tailored for data imputation, leveraging both global spatial and temporal dependencies. Empirical experiments conducted on diverse publicly available datasets substantiate the efficacy of ST-Bi-LSTM-VAE. The results obtained consistently demonstrate that proposed method surpasses baseline techniques in maintaining pattern, structure, and trend across datasets even at 50% missing gap for imputation task with 4.91% performance improvement in case of Intel Berkley Research Laboratory (IBRL) dataset and 3.5% on PRSA dataset.

Keywords: Internet of Things (IoT), Data imputation, Multivariate data, Spatial-temporal correlation, Variational auto encoder (VAE), Data quality.

1. Introduction

The advancement of Internet of Things (IoT) has facilitated an extensive number of applications, encompassing, but not confined to, health care, food traceability, environmental monitoring, and smart transportation and infrastructure [1, 2]. The IoT comprises a large number of intelligent sensors distributed across numerous networks, collecting data. IoT systems can be highly helpful in various contexts, but they may also face challenges due to high demands, aging or insufficient infrastructure, and harsh operating conditions caused by climate changes [3]. Depending on the area, severe events can include flooding, winds, snow squalls, extreme temperatures, and other weather conditions that can negatively impact the operation of some infrastructure components and hosted IoT devices.

Subsequently, the data produced by these devices is riddled with severe anomalies and data gaps. Nevertheless, every intelligent decision-making technique operates under the presumption that comprehensive data is anticipated. In the IoT and its applications, such as intelligent transportation systems [4], smart health, environmental monitoring [5], and energy management are all susceptible to missing data for a variety of reasons, including faulty sensors and unstable network communications [6]. The existence of absent values within the dataset disrupts the data's pattern, structure, seasonality, and trend, thereby impeding the production of precise information that is necessary for making informed decisions [7].

Data obtained from IoT devices exhibit two significant characteristics: temporal and spatial features. When data is collected by a single sensor, it is typically arranged and stored chronologically, forming a time series. However, considering the device itself, there can be spatial relationships among groups of devices, forming a network with a non-Euclidean topological structure. Consequently, sensors within such networks may collect time series data that demonstrate strong spatial dependencies. These data, combining both temporal and spatial aspects, constitute spatiotemporal data.

For instance, when monitoring rainfall in geological disaster-prone areas, devices deployed randomly within the same mountain or forest can be grouped based on their non-Euclidean spatial relationships. It becomes reasonable to fill missing values by considering both the historical data collected from the same device (temporal feature) and the current data from spatially correlated devices (non-Euclidean spatial feature).

Existing models demonstrated the importance of incorporating spatial information for imputing missing values in traffic data. However, they have primarily focused on utilizing spatial data from neighbouring locations, overlooking the potential benefits of leveraging non-Euclidean (global) spatial features combined with temporal features to a fuller extent. The proposed method ST-Bi-LSTM-VAE (Spatio-Temporal Bidirectional Long Short-Term Memory based Variational Auto-Encoder) utilised non-euclidian spatial correlations with temporal correlations by incorporateing two distinct variations of Variational Auto-Encoders (VAEs) designed to address distinct tasks: the SF-VAE (Spatial Features VAE) and the TF-VAE (Temporal Feature VAE). In more depth, we present the following technical contributions:

1. SF-VAE for adjacency matrix: SF-VAE is a novel application of the VAE framework. This pioneering approach utilizes VAE to construct an adjacency matrix suitable for network of IoT devices. It uses Graph Convolutional Network (GCN) to capture non Euclidian spatial dependencies.

2. TF-VAE for data imputation: Leveraging the spatial characteristics captured by the GCN, TF-VAE (Temporal Feature VAE), a multi-head Bi-LSTM-based VAE captures temporal features while simultaneously fulfilling data imputation tasks. Notably, both the Bi-LSTM hidden state (h) and the VAE latent variable (l) sampling are ordered in time, thereby endowing the TF-VAE with the capacity to extract temporal correlations. Furthermore, we apply Linear Normalization Flow to convert the Gaussian/Normal distribution of l into a randomized space, that improves the resilience of the network.

3. Comprehensive Spatial-Temporal Approach: Our stochastic-based methodology combines the advantages of SF-VAE and TF-VAE to achieve remarkable dynamic generality prowess for individual variability, while simultaneously performing data imputation based on universal spatial and temporal correlations. This augmentation enhances the realism of imputed values.

4. Experiments and Comparisons: The superior performance of our ST-Bi-LSTM-VAE method compared to numerous state-of-the-art alternatives.

The amalgamation of these components leads to a powerful approach capable of precise data imputation, with promising generative capabilities validated through extensive experimentation. Further the existing literature regarding data imputation in IoT is discussed in section 2. Sections 3 provided with preliminaries required for understanding the proposed method which is described in section 4. Section 5 presented with experiment setup, baseline methods for comparisons, evaluation metrics and dataset description. Section 6 describes the performance evaluation followed by conclusion in section 7.

2. Related works

The Intelligent Internet of Things (IIoT) is comprised of a large quantity of intelligent sensors that are distributed across many networks. These sensors collect data that demonstrate unforeseen and unusual deviations. In general, approaches to address missing data in time series can be statistical methods, machine learning based, deep learning based and others. The integration of intelligent approaches, such as machine learning, deep learning, and the Internet of Things (IoT), is a key aspect of the IIoT [8, 9]. The literature pertaining to incomplete time series data modelling is presented in this section.

To compensate for missing values, statistical methods utilize statistical values; for instance, the mean filling approach [10] and fuzzy-rough nearest neighbours [11] are two examples. Nevertheless, statistical approaches are inadequate for imputation of missing data in case of IoT, as these methods may not utilize temporal insight. The utilization of machine learning methods has made significant strides in data filling applications, which primarily involve determining the distribution of the original data and constructing an imputation model. In [33], Lightweight Window Portion-based Multiple Imputation (LWPMI) based on multivariate

variables, correlation, data fusion, regression, and multiple imputation was proposed using Lightweight Gradient Boosting Machine (LGBM) regression. KNN imputes missing data using the average of values from k neighboring nodes located around the absent sample. Additionally, matrix factorization was implemented to impute absent values. While conventional machine learning approaches were often effective in restoring missing values across various domains, they lack the capability to account for the temporal relationships between two observations [13]. Consequently, such approaches were unsuitable for IoT large data environments.

The emergence of Generative Adversarial Networks (GANs), including GAIN [14], MisGAN [13], and others, introduced a method for imputing missing data values. By utilizing incomplete time series, a GAN-based method [13], was intended to discover the overall distribution of a multivariate time series dataset. ImputeGAN, proposed in [34] was an iterative GAN based imputation for multivariate data imputation for various types of data. Utilizing a new complete sample produced by the auto-encoder and GRUI, E2GAN was suggested as a method to impute incomplete time series [15]. Despite the notable efficacy of these models in imputation, they continue to be discarded as a data preprocessing stage in assignment to classification tasks. Additionally, they were not optimized at the same time as training the classifier, which will lead to unsuitable outcomes [16].

Traffic data is a collection of sequential data gathered over a period of time. Recurrent Neural Networks (RNNs) handle sequence data with general efficacy [17] due to the implementation of special gate mechanisms and the upkeep of chain-like structures. Leveraging a basic RNNs, such as a Gated Recurrent Unit (GRU) or Long Short-Term Memory (LSTM), was considered as the simplest method for imputation [18]. But this usually results in less-than-ideal behavior because the model learns biassed parameters at first because it substitutes in missed data with predetermined data [19]. To enhance the network's resilience and facilitate the ability to manage data that is not available, [20], [21] introduced innovative network architectures based on LSTM. The masking vector was incorporated into a regular LSTM structure in order to enhance the modelling of the missing patterns, as described in [20]. In accordance with this, [21] incorporated an imputation module within the LSTM architecture, whereby the absent values at the present time step were replaced with values deduced from the preceding cell and hidden states. An alternative approach involves aligning the attention mechanism with a modified version of a recurrent neural network [22]. Despite their moderate level of expressiveness, these approaches fail to consider the geographical interdependencies present in traffic data. For data imputation, [35] proposed Time-Recurrent Variational AutoEncoder ODE (TRVAEODE) model using a time-aware LSTM encoder and neural ordinary differential equations (ODEs). This method did not utilise spatial correlations. Dynamic Adaptive Network-Based Fuzzy Inference System (D-ANFIS) was proposed in [36] for medical data imputation in Internet of Medical Things (IoMT). D-ANFIS did not consider spatial and temporal correlations.

Graph Neural Networks (GNNs) have recently demonstrated remarkable efficacy in modelling non-Euclidean data, specifically graphs [23], have achieved good results in smart transportation. [24] created a model using a GCN to simultaneously estimate traffic and fill in missing data. The model utilized weighted adjacency matrix based on distances to represent spatial relationships. In their study, [25] introduced a spatial interactive GCN network for the job of imputing. [26] developed a GraphSAGE model to gather spatio-temporal data from a graph created using correlation coefficients of past values. This was done since a fixed graph based on distance was unable to accurately represent the variations in spatial correlations over time. A heterogeneous graph-based GCN model in [27] constructed a multigraph by utilizing spatial and past data depict the intricate interrelationships between portions of road. A further kind of GNN, known as GAT, was created to tackle the issue of incomplete traffic data in [28]. GAT integrates the attention mechanism. While these methods primarily concentrate on representing geographical relationships, they are simplistic and depend on having a substantial amount of past data. In their study, [29] presented a spatial context sensing model that utilizes information from nearby sensors to reconstruct traffic data.

These models illustrate the utility of spatial information in the process of imputation for traffic data. Nevertheless, their emphasis has been on utilizing local spatial data obtained from nearby locations, neglecting to fully exploit global spatio-temporal information.

3. Preliminaries

3.1 Problem statement

The data collected in this study are sourced from diverse sensors integrated into various devices. These sensors amass data and arrange it in the form of time series. In this research, Data correspond to spatial and temporal data gathered by numerous sensors within the network of devices. It can be expressed as $S^{mon} = \{S_1^{mon}, S_2^{mon}, S_3^{mon}, \ldots\ldots S_T^{mon}\}$ here, T represents the length of time series.

The dimensions of $S_t^{mon} = s_t^{mon-1}, s_t^{mon-2}, s_t^{mon-3}, \ldots\ldots s_t^{mon-X}$ encompass X*Y, and each s_t^{mon-X} contains Y values. These values signify the presence of X devices collectively, and within each device, Y diverse sensors are operational. The core objective of this paper revolves around imputing missing values within S^{mon}. This imputation is executed by utilizing existing data. In this article, our objective is to minimize the disparity between imputed data and actual data, leading us to formulate the imputation problem as follows:

$\min_{\hat{s}} \sum_{i=0}^{X} \sum_{j=0}^{Y} \left| \hat{s}_i^j - s_i^j \right|$, where $\hat{s}_i^j \epsilon \hat{S}$ is the imputation of $s_i^j \epsilon S^{mon}$. Table 1 provides the list of notations for understanding the proposed method.

3.2 Missing data types

There are three distinct categories that encompass missing data situations: Missing Completely at Random (MCAR), Missing at Random (MAR), and Missing Not at Random (MNAR). MCAR pertains to cases where the missingness is unrelated to any observed or unobserved data, akin to the randomness of a coin flip. MAR involves missingness that can be elucidated by examining available data. For instance, missing spouse age data might be linked to the marital status data. On the other hand, MNAR arises when the absence of data is contingent on attributes that are either unobserved or connected to the missing attribute itself. This scenario is evident in instances such as prolonged device failures. Our primary focus is on addressing the first two types of missing data, as imputing missing values in MNAR scenarios holds lesser significance.

3.3 Variational auto encoder (VAE)

A Variational Autoencoder (VAE) is a type of generative model that combines techniques from both autoencoders and variational inference to learn a latent space representation of data and generate new data samples. It is widely used in tasks like data generation, compression, imputation and representation learning.

In a VAE, the main idea is to assume that the observed data is generated from a latent variable and to learn a probabilistic model that can capture the underlying distribution of this latent space. The key components of a VAE are the encoder, the latent space, and the decoder.

3.3.1. Encoder (recognition model)

The encoder maps input data (x) to a distribution in the latent space using Equation 3.1. This distribution is typically a Gaussian distribution characterized by a mean M and a standard deviation SD. The encoder network is denoted as $(q_\phi(z|x))$, where (z) represents the latent variable and (ϕ) represents the encoder's parameters.

$$q_\phi(z|x) = \mathcal{N}\left(M_\phi(x), SD_\phi(x)^2\right) \quad (1)$$

In the Equation 3.1, the symbol $\mathcal{N}$ represents the normal distribution. In statistics and probability theory, the normal distribution is a commonly used probability distribution also known as the Gaussian distribution. It's characterized by its bell-shaped curve and is fully described by its mean $M_\phi(x)$ and variance $SD_\phi(x)^2$. So, the equation is specifying that the distribution of the generated data x given the latent variable z follows a normal (Gaussian) distribution with mean $M_\phi(x)$ and variance $SD_\phi(x)^2$. This is a fundamental assumption in the VAE framework, where the decoder generates data samples by sampling from this normal distribution.

3.3.2. Latent space

The latent variable (z) follows the distribution output by the encoder. This distribution in Equation 3.2 is then used to sample a latent vector.

$$Z \sim Q_\Phi(Z|X) \quad (2)$$

3.3.3. Decoder (generative model)

The decoder takes a sample from the latent space (z) and maps it back to the data space using Equation 3.3. The goal is to reconstruct the input data as accurately as possible. The decoder network is denoted as $(p_\theta(x|z))$, where (θ) represents the decoder's parameters.

$$P_\Theta(X|Z) = \mathcal{N}(X|M_\Theta(Z), SD_\Theta(Z)^2) \quad (3)$$

3.3.4. Loss function

The loss function for a VAE includes two components: a reconstruction loss that measures how well the generated data matches the input data, and a regularization term that encourages the latent space to follow a prior distribution (usually a standard Gaussian distribution). The reconstruction loss is typically a measure like the mean squared error (MSE) or the binary cross-entropy (BCE) between the input data and the reconstructed data. The regularization Kullback-Leibler (KL) divergence measures the difference between two probability distributions. In the context of a VAE, it quantifies how much the encoder's distribution deviates from the prior distribution. Mathematically, the KL divergence between two distributions (q(z|x)) and (p(z)) is defined as shown in Equation 3.4:

$$KL\big(q(z|x)||p(z)\big) = \sum_i q(z_i|x)\,\mathrm{LOG}\frac{q(z_i|x)}{p(z_i)} \quad (4)$$

The overall loss as shown in Equation 3.5 function is a combination of these two terms:

Table 1. List of notations

Notation	Meaning
S^{mon}	Monitor data that is to be imputed
S_i^{mon}	Data at Node i
$q_\phi(z\|x)$	Encoder network
z	Latent variable
ϕ	Encoder parameters
$M_\phi(x)$	Mean of x
$SD_\phi(x)^2$	Variance of x
$p_\theta(x\|z)$	Decoder network
θ	Decoder parameters
$(q(z\|x))\ \&(p(z))$	Data Distributions
L(x)	Reconstruction Loss
Adj^{mon}	Adjacent matrix
FM^{mon}	Feature Matrix
D_{KL}	Kullback-Leibler (KL) divergence
$JS(z1, z2)$	JS (Jensen-Shannon) between latent variables $z1, z2$
α	weight parameter
l	hidden layer
W	parameter matrix
H	Hidden-feature matrix
I	Identity matrix

$$\mathcal{L} = \text{Reconstruction Loss} + \text{KL Divergence Regularization} \quad (5)$$

where Reconstruction Loss L(x) can be calculated using Equation 3.6

$$L(x) \approx \frac{1}{n}\sum_{i=1}^{n}\Big(\mathrm{LOG}\,p\left(x|z^i\right)\Big) + \mathrm{LOG}\,p\left(z^i\right) - \mathrm{LOG}\,q\left(z^i|x\right) \quad (6)$$

The VAE is trained by optimizing this loss function with respect to the encoder and decoder parameters. In summary, a VAE uses an encoder to map input data to a latent distribution, samples from this distribution, and then employs a decoder to generate data samples. The model is trained to minimize the reconstruction loss while also encouraging the latent space to follow a specific distribution through the KL divergence term. This encourages the VAE to generate meaningful and continuous latent representations of the input data.

4. Proposed approach

Data pre-processing task and imputation task are the two main parts of the ST-Bi-LSTM-VAE process, which is shown in the Figure 1.

In the Data pre-processing phase, we employ SF-VAE to handle the data S^{mon}, represented as $S_1^{mon}, S_2^{mon}, S_3^{mon}, \ldots, S_T^{mon}$, and transform it into the adjacent matrix Adj^{mon}. Additionally, we are required to reshape the this into the Feature Matrix FM^{mon}. These two elements serve as crucial inputs for the subsequent data imputation stage.

Next, the adjacency matrix Adj^{mon} and the Feature Matrix FM^{mon}. are introduced into the imputation process, to address missing values within the data. The data imputation workflow can be further divided into 3 phases: encoder phase, feature mapping/ transformation phase, and decoder phase. TF-VAE enhances both the encoder and decoder stages, whereas feature/ attribute transformation utilizes normalization flow.

4.1 SF-VAE

SF-VAE plays a crucial role in preprocessing the input data into an adjacency matrix Adj^{mon}, a necessary initial input for utilizing GCN to extract non-Euclidean spatial relationships within the data in TF-VAE. Many state-of-art studies made the assumption that the structural arrangement of the graph is already established and provided. However,

this assumption can often be challenged in practical situations. In contrast to this assumption, our approach involves the utilization of SF-VAE along with JS divergence to derive the adjacency matrix that encapsulates the non-Euclidean spatial relationships within the network of devices. The core concept of proposed methodology revolves around leveraging data symmetry to quantify the spatial connections among devices.

To achieve this, we commence by inputting X multivariate time series, denoted as $S^{mon-1}, S^{mon-2}, \ldots, S^{mon-X}$, originating from X distinct devices, into X separate SF-VAEs. This process yields distinct representations of a hidden variable z, each of which can effectively characterize the structure of $S^{mon-x} = s_1^{mon-x}, s_2^{mon-x}, s_3^{mon-x}, \ldots, s_T^{mon-x}$ belongs to various devices once the training of model is completed. After that, we estimate the JS divergence among these hidden variable distributions and use it to create the adjacency matrix for the device network.

In contrast to TF-VAE, only integrate multi head attention based Bi-LSTM integrated into the SF-VAE network as shown in Figure 3. This choice is made because time series data from various devices are directed to distinct SF-VAEs, making the sole utilization of temporal features sufficiently. Furthermore, for a given device, all timestamp values share a common mean denoted as M and a common variance represented as V. This choice is intentional, aiming to encapsulate the overall data pattern at the device level. Consider S1 and S2, 2 multivariate time series that come from different nodes. We've obtained feature distributions, FM1 and FM2, through the use of SF-VAEs. Specifically, FM1 can be represented as (M1, V1), and FM2 as (M2, V2). Now, we define a new distribution, FM3, as the average of FM1 and FM2, calculated using Equation 4.1:

$$\mathrm{FM3} = \frac{(\mathrm{FM1} + \mathrm{FM2})}{2} \text{ and } \mathrm{FM3} \approx \frac{(\mathrm{M1} + \mathrm{M2})}{2}, \mathrm{FM3} \approx \frac{(\mathrm{V1} + \mathrm{V2})}{2} \tag{7}$$

Subsequently, The Jensen-Shannon (JS) divergence between FM1 and FM2 using Equation 4.2. In this equation, D_{KL} represents the Kullback-Leibler (KL) divergence (Equation 4.3), defined as

$$\mathrm{JS}(z1, z2) = \frac{1}{2}\mathrm{KL}\left(\mathrm{FM1} \parallel \frac{\mathrm{FM1} + \mathrm{FM2}}{2}\right) + \frac{1}{2}\mathrm{KL}\left(\mathrm{FM2} \parallel \frac{\mathrm{FM1} + \mathrm{FM2}}{2}\right) \tag{8}$$

$$\mathrm{D_{KL}}(P//Q) = \sum_{s \in S} \mathrm{P(s)} \log\left(\frac{1}{\mathrm{Q(s)}}\right) + \sum_{s \in S} \mathrm{Q(s)} \log\left(\frac{1}{\mathrm{P(s)}}\right) \tag{9}$$

The JS (Jensen-Shannon) similarity metric can be used to quantify the degree of resemblance between two time series that originate from different devices, denoted as JS (z1, z2). Correspondingly, the adjacency matrix Adj^{mon} is computed using above Equation 4.4.

$$\mathrm{Adj_{ij}}^{\mathrm{mon}} = \begin{cases} 0, & \text{if} \mathrm{JS}(z1, z2) > \beta \\ 1, & \text{otherwise} \end{cases} \tag{10}$$

As shown in Figure 2, the data collected from sensors inside each device is organized in one row, while data from other devices is placed in distinct rows, to transform the input data into FM^{mon}.

4.2 TF-VAE

The fundamental structure of the VAE comprises two components: the encoder and the decoder. In the context of TF-VAE, we adopt a combination of GCN and multi-head attention based Bi-LSTM to implement the encoder, while the decoder exclusively employs the multi-head attention based Bi-LSTM. This choice of architecture serves specific purposes: the GCN is leveraged to capture non-Euclidean spatial features, while the attention based multi-head Bi-LSTM is employed to handle temporal features. Consequently, the model effectively captures spatio-temporal information from IoT data.

It is important to note that TF-VAE diverges from traditional autoencoders by incorporating a more generative generation model. This introduces a higher level of stochasticity into the process of filling missing values, rendering the imputed data more realistic in its representation. Figure 4 provides a comprehensive view of the encoder component within TF-VAE. The process involves sending Adj^{mon} (adjacent matrix) and AM^{mon} through numerous layers of GCN to capture non-Euclidean spatial features inherent in data. Subsequently, these features are directed into the multi-head attention based Bi-LSTM to acquire knowledge about

temporal dependencies, device by device. This sequence of operations yields the spatio-temporal feature representation, denoted as hf_0^{mon} through hf_t^{in}. Parameter α is weight parameter that require training alongside the di-vae model as shown in equation 4.5.

$$hf^{comb} = (\alpha * hf^{mon}) \tag{11}$$

In contrast to the conventional encoder used in VAE, the TF-VAE introduces a novel approach to incorporate temporal dependencies not only into the hidden states hf_0^{comb} through hf_T^{comb} of the multi-head attention based Bi-LSTM but also into the latent variables z_0 through z_T. In the traditional VAE framework, every z_T is independently drawn from a Gaussian Distribution, which implies that there are no inherent temporal correlations among them.

To address this limitation, we take a different route by concatenating h_t^{comb} and z_{t-1} before feeding them into the Fully Connected Layer (FCN) for the purpose of learning the distribution parameters (M_t and V_t) of z_t, as illustrated in Equation 4.7. This concatenation of h_t^{comb} and z_{t-1} allows the sampled z_t to possess temporal relationships with the previous z_0 through z_{t-1}, thereby incorporating valuable temporal context into the latent variable generation process.

$$M_t = w^M\left[z_t, hf_t^{comb}\right] + b^X,\ \sigma_t = Re\,LU\left(w^\sigma\left[z_{t-1}, hf_t^{comb}\right] + b^\sigma\right) \tag{12}$$

In the Decoder of TF-VAE, we exclusively employ the multi-head attention based Bi-LSTM to decode the latent variables z_0 through z_T and reconstruct the original Monitor Data, denoted as S^{mon}. This process effectively fills in all the missing values present in the initial input data.

4.2.1. GCN

Starting with GCN, we initiate the process by inputting the adjacent and feature matrices derived from data into GCN. In a traditional GCN network, we compute each hidden layer (l) using Equation. 4.7, where 'Adj' represents the adjacent matrix, and 'W' corresponds to the parameter matrix which undergoes training and 'H' is the hidden-feature matrix. Additionally, we set H_0 to be the Feature Matrix (FM_{mon}).

$$Hf^l = \sigma\left(\text{Adj}.\, H^{l-1}W^{l-1}\right) \tag{13}$$

Nonetheless, when utilizing Equation.4.7, two significant challenges persist. Firstly, it fails to propagate a node's specific features to the subsequent layer by simply multiplying the 'Adj' with 'H'. Secondly, there's the issue of potential rapid growth in the values of the feature map as layers are successively extended.

To address these issues, a solution presented in [20] proposes replacing 'Adj' with an adjusted adjacency matrix 'Adj' = I + Adj, where 'I' signifies an identity matrix. Furthermore, an additional spectral normalization scheme is introduced. The refined equation is expressed as Equation 4.8:

$$Hf^l = \sigma\left(D^{\frac{-1}{2}}\text{Adj}'D^{\frac{-1}{2}}Hf^{l-1}W^{l-1}\right),\quad D_{ii} = \sum_j \text{Adj}'_{ij} \tag{14}$$

The feature maps produced by the GCN are partitioned into separate groups based on the specific devices they pertain to. These groups are subsequently processed by a multi-head attention based Bi-LSTM to capture temporal features. In this configuration, each head of the multi-head attention based Bi-LSTM corresponds to a distinct device. The rationale behind employing a multi-head attention based Bi-LSTM in our study is twofold:

Firstly, data originating from various devices often exhibit significant differences in their inherent characteristics. Hence, it is not recommended to input such varied data into a unified set of Bi-LSTM that utilizes the identical parameter matrix.

Secondly, the utilization of a multi-head attention based Bi-LSTM structure introduces greater adaptability to our network. This is particularly relevant in the context of the Internet of Things (IoT), where the arrangement and configuration of devices can frequently change due to installations, removals, or modifications. It's noteworthy that our approach differs from the conventional practice of utilizing only the last timestamp's hidden state, ($hf_{T_{comb}}$). Instead, we incorporate hidden states from multiple timestamps, ($hf_{0_{comb}}$) to ($hf_{T_{comb}}$) (where (T) is the maximum timestamp), as the hidden states within our auto-encoder. This decision is rooted in findings presented in [21], which indicate that when employing Recurrent Neural Networks (RNN) as auto-encoders, leveraging hidden states from multiple timestamps tends to yield superior results compared to relying solely on the last timestamp's hidden state.

4.2.2. Multi-head attention based BI-LSTM

After receiving the input from GCN, proposed model leverages a multi-head self-attention mechanism to process that data. This mechanism enables the model to collectively focus on various aspects of the input representations and determine the degree of importance each element carries in generating the output.

For a sequential feature set $S \in R^{p \times q}$ (with sequence length p and dimensionality q), the self-attention mechanism initially projects these features using independent linear projection functions (LN) into queries $Q \in R^{p \times q_k}$, keys $K \in R^{p \times q_k}$, and values $V \in R^{p \times q_v}$. Subsequently, it computes a specific operation to determine the attention weights assigned to the values. This operation involves calculating the dot product between the query Q and all keys K and then normalizing the result by $\sqrt{q_k}$, where q_k represents the dimension of queries and keys. Finally, a softmax function is applied to the normalized result, as illustrated

in Equation 4.9. This process can be repeated in parallel multiple times.In essence, this self-attention mechanism allows the model to efficiently capture and weigh the significance of different elements within the input sequence, enhancing its ability to produce meaningful outputs.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(QK^T/\sqrt{d_k}\right)V \quad (15)$$

The Bidirectional Long Short-Term Memory (Bi-LSTM) model, is an extension of LSTM models that involves the application of two LSTM layers to the input data.

Allowing it to capture past and future context This architecture incorporates LSTM units that operate in both forward and backward directions, information simultaneously. This dual application of LSTM layers enhances the model's ability to learn long-term dependencies effectively, as it avoids retaining redundant context information and ultimately leads to improved model accuracy.

The Bi-LSTM model comprises several essential components. The input gate, denoted as (i_t), controls the flow of information into the memory cell. On the other hand, the output gate, denoted as $(\underline{o_t})$, governs how the state of the memory unit affects the network's output at the current time step. Additionally, the forget gate, represented as $(\underline{f_t})$, determines the duration for which existing information is preserved.

At each time step (t), the value of the memory unit $(\underline{C_t})$ undergoes an update process. This update involves filtering the previous information using $((f_t \cdot \underline{C_{t-1}}))$ and incorporating candidate information computed as $((i_t \cdot \underline{\tilde{C}_t}))$, as described by the Equations 4.10, 4.11, 4.12 and 4.13:

$$\underline{C_t} = f_t \cdot \underline{C_{t-1}} + i_t \cdot \underline{\tilde{C}_t} \quad (16)$$

$$\underline{\tilde{C}_t} = tan\,h\left(\underline{W_c} \cdot \left[\underline{h_{t-1}}, \underline{x_t}\right] + \underline{b_c}\right) \quad (17)$$

$$i_t = \sigma\left(\underline{W_i} \cdot \left[\underline{h_{t-1}}, \underline{x_t}\right] + \underline{b_i}\right) \quad (18)$$

$$\underline{f_t} = \sigma\left(\underline{W_f} \cdot \left[\underline{h_{t-1}}, \underline{x_t}\right] + \underline{b_f}\right) \quad (19)$$

In the context of this model, σ represents an activation function, typically the sigmoid function. Following the update of the memory unit, the calculation of the current hidden layer (h_t) is determined based on the current output gate (o_t), as specified in Equation 4.14 and 4.15.

$$\underline{o_t} = \sigma\left(\underline{W_o} \cdot \left[\underline{h_{t-1}}, \underline{x_t}\right] + \underline{b_o}\right) \quad (20)$$

$$\underline{h_t} = \underline{o_t} \cdot \tan\text{h}\left(\underline{C_t}\right) \quad (21)$$

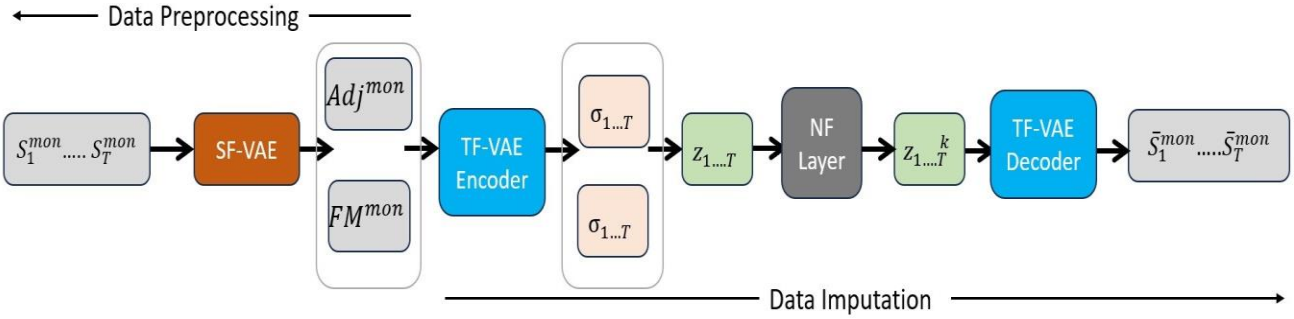


Figure 1: Workflow of ST-Bi-LSTM-VAE including data preprocessing and imputation

Sensor 1 Value	Sensor 2 Value	Sensor 3 Value	Sensor 4 Value	Device i
Sensor 1 Value	Sensor 2 Value	Sensor 3 Value	Sensor 4 Value	Device i+1
Sensor 1 Value	Sensor 2 Value	Sensor 3 Value	Sensor 4 Value	Device i+2
Sensor 1 Value	Sensor 2 Value	Sensor 3 Value	Sensor 4 Value	Device i+3
Sensor 1 Value	Sensor 2 Value	Sensor 3 Value	Sensor 4 Value	Device i+4

Figure 2: Feature matrix arrangement of Monitor data

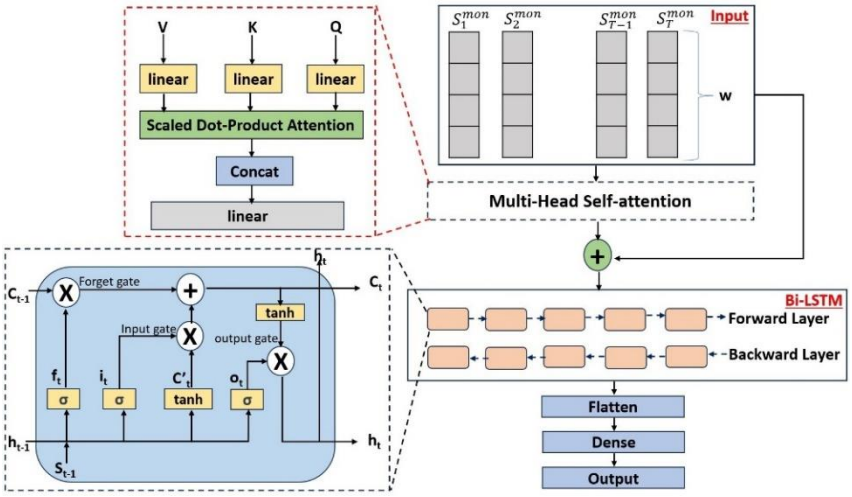


Figure 3: Encoder In SF-VAE

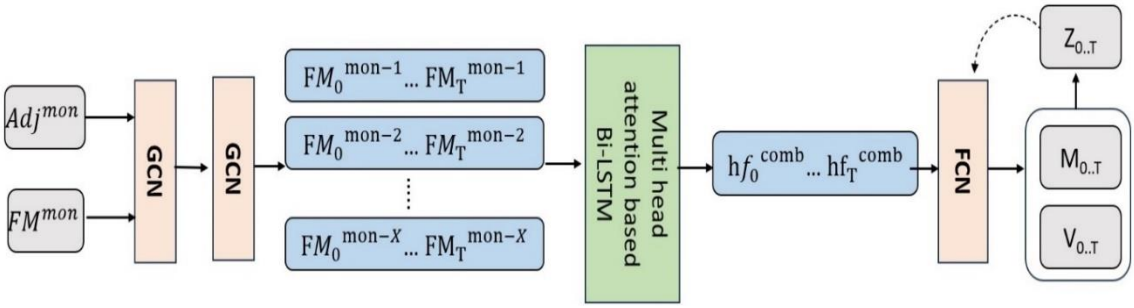


Figure 4: Encoder In TF-VAE

Optimizing the hyperparameters of deep learning models is a crucial step in constructing an effective DL model that can achieve peak performance. In this study, we undertook a comprehensive exploration of various hyperparameters for fine-tuning purposes. We employed a random search technique to identify the optimal combination of hyperparameters for our neural network.

The following hyperparameters were determined to be optimal for our model:

- Optimizer type: Adam [20],
- Learning rate: (1×10^{-5}),
- Number of units: 64,
- Number of heads: 4,
- Activation function: ReLU.

guarantees that the output of each GCN layer aligns with the format outlined in Equation 4.4. In addition, the latent variable z is assigned a value of 8 and the length of NFs is 4.

These specific hyperparameter values were found to yield the best overall performance in terms of both accuracy and computational efficiency. Where the under-bars represent backward path.Using Y' to represent the predicted system states in the output sequence of the second bidirectional LSTM layer, we present a system state prediction setup using a bi-directional LSTM (Bi-LSTM) depicted in Figure 3. The Bi-LSTM consists of two LSTM layers. Between these layers, the

outcomes from both the prediction and smoothing operations are combined and passed on to the subsequent LSTM layer to achieve varying levels of representation. Following the second LSTM layer, the ultimate prediction at each time step is generated by integrating the outcomes from both forward and backward paths.

4.3 Normalization flow

The hidden variable 'z' is drawn from the distribution q(z|x), a common assumption in traditional VAE is that this distribution follows a Gaussian distribution. However, relying on this assumption can diminish the network's robustness, as q(z|x) may not consistently conform to a Gaussian distribution.

To address this problem, we utilize a method known as normalization flow (NF) to transform the distribution q(z|x) into a conventional distribution through a series of reversible mappings. NF applies a set of reversible mappings $g^1, g^2, \dots, g^K$ to build a potentially intricate distribution $z^K = g^K \circ g^{(K-1)} \circ \dots \circ g^0(z^0)$, resulting in the following equation 4.16:

$$\log pz^k = \log pz^{(k-1)} - \log \left| \det \frac{\partial g^k}{\partial z^{k-1}} \right| =$$

$$\log pz^k = \log pz^{(k-1)} - \log \left| \det \frac{\partial g^k}{\partial z^{k-1}} \right| \quad (22)$$

Specifically, our choice falls on Linear Normalization Flow (Linear NF) due to its capability to articulate various forms of correlation between dimensions and IoT data. Furthermore, it boasts a straightforward and uncomplicated structural design. The definition of Linear NF is as follows:

In the context provided, we have parameters $A \in R^D (D * D)$ and $b \in R^D$ that require training within a neural network. Notably, A represents an invertible matrix. It's worth mentioning that the determinant of the Jacobian matrix can be straightforwardly computed as det(A), with a computational complexity of $O(D^3)$, similar to the calculation of the matrix's inverse using Equation 4.17.

$$g(z) = Az + b \quad (23)$$

Conventionally, convolutions are known for their ease of inverse computation, while the determinant computation is less intuitive. To address this, [23] introduced a novel approach using 1-Dimensional convolutions to efficiently calculate the det (determinant) of the Jacobian matrix. We adopted their method to optimize linear normalizing flows (NF), as shown in Equation 4.18:

$$g(z) = z + u \odot h\big(\text{conv}(z, w)\big) \quad (24)$$

In Equation 4.18:

- $w \in R^K$ represents the filter used in the 1-Dimensional convolution.
- h is the activation function.
- $u \in R^d$ is a vector used to adjust the dimensions of the output derived from 'h'.

By incorporating these techniques, we aim to enhance the efficiency and effectiveness of linear normalizing flows within our neural network.

5. Proposed approach

The performance enhancement of the suggested model has been assessed and validated through a comparison with existing imputation models. To quantitatively assess the performance of the ST-Bi-LSTM-VAE model in imputation, this study reports Mean Squared Error (MSE), R- Squared Error R^2 Error), and Mean Absolute Error (MAE), comparing

it with various baseline methods MDIMTFL, ImputeGAN, TRVAEODE, and D-ANFIS.

5.1 Experimental setup

The proposed method evaluated on Intel lab dataset. The experiments utilise Python 3.8.5 for coding purposes. PyTorch is chosen as the foundation for implementing our ST-Bi-LSTM-VAE, a deep learning model that incorporates GCN, VAE, and Bi-LSTM, along with other comparison techniques.

In the ST-VAE section, the hidden state h_{0-t} for all timestamps in the Multi-head Bi-LSTM corresponds to a consistent average value denoted as l and variance r^2. This information is visually represented in Figure 2. The size of the latent variable z is explicitly set to be smaller than the output h of the Bi-LSTM. A hyperparameter search is utilized to attain optimal results across the IBRL dataset. The specific values assigned to z and h are 4 and 8, respectively.

In the DI-VAE component, the hidden state h_{0-t} of all timestamps in the Multi-head Bi-LSTM is delivered to a single Fully Connected Network (FCN). However, each hidden state receives distinct mean and variance values, as depicted in Figure 2. In the case of Multi- head Bi-LSTM, the no. of

Bi-LSTMs corresponds to the no. of devices. The hidden state dimension of the output from the Data Bi-LSTM is set to 16. Concerning GCN, the lone hyperparameter is the structure of the weight matrix W. It is set to k*k, with k representing the count of sensors within a single device. This configuration

5.1.1. Intel dataset

The Berkeley Research Lab at Intel (IBRL) has set up a wireless monitoring network with 54 nodes to collect information about the area around the lab. These nodes, which are outfitted with sensors for light, temperature, humidity, and voltage, gather information on a range of environmental variables every 31 sec. The sensor data that was collected can be found in an open repository that can be used for study. Randomly adding missing sub-sequences of length were done on purpose throughout the whole dataset from Intel's Berkeley Research Lab website to show how well the suggested method works. The suggested method and the values that were assumed from other algorithms were then checked against the real values.

5.1.2. PRSA dataset

Recordings of air pollutants gathered (hourly) from Beijing, China's Twelve national air-quality monitoring stations are included in the PRSA dataset [24]. The dataset covers the time between March, 2013, to 28th February, 2017. It includes data from six types of sensors measuring pollutants PM2.5, PM10, SO_2, NO_2, CO and O_3.

5.2 Data pre-processing

Missing values in the original dataset are initially replaced using mean imputation. The dataset is subsequently split into two partitions, with one portion allocated for training and the other portion reserved for testing. During the testing step, certain values are randomly omitted prior to conducting a comparative experiment, and the respective places of the omitted values are annotated. In the IBRL dataset missing gaps were created with rates ranging from 0.1 to 0.5, enabling an assessment of the efficacy of prominent strategies in ST-VAE.

It is important to mention that in practical IoT situations, if over 50% of the data gathered by a sensor is missing, the sensor is deemed defective and should be substituted. When the missing rate surpasses 50% in such circumstances, populating the missing data loses significance. The purpose of this setup is to test and evaluate the effectiveness of different strategies in dealing with situations where data is missing specifically in the context of ST-VAE.

5.3 Evaluation metrics

5.3.1. Mean absolute error (MAE)

It measures the average absolute difference between the imputed values $\left((\text{Imputed}_{ij})\right)$ and the true (actual) values $\left((\text{Actual}_{ij})\right)$ across all instances and variables using Equation 5.1.

$$MAE = \frac{1}{mn}\sum_{i=1}^{n}\sum_{j=1}^{m}\left|\text{Actual}_{ij} - \text{Impute}_{ij}\right| \quad (24)$$

5.3.2. Mean squared error (MSE)

The Mean Squared Error (MSE) measures the average of the squared differences between the imputed values $\left((\text{Imputed}_{ij})\right)$ and the true (actual) values $\left((\text{Actual}_{ij})\right)$ across all instances and variables using Equation 5.2.

$$MSE = \frac{1}{mn}\sum_{i=1}^{n}\sum_{j=1}^{m}\left(\text{Actual}_{ij} - \text{Imputed}_{ij}\right)^2 \quad (25)$$

5.3.3. R^2 squared error

In the context of multivariate imputation, the R^2 error can be adapted to evaluate the performance of imputation models. For multivariate data, where each observation has multiple variables, R^2 can be calculated separately for each variable or as an overall measure for the entire imputation process. The formula for R^2 error in the context of multivariate imputation can be calculated using Equation 5.3:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}\left(Actual_{ij} - \text{Mean(Actual)}_j\right)^2}{\sum_{i=1}^{n}\sum_{j=1}^{m}\left(Actual_{ij} - Imputed_{ij}\right)^2} \quad (26)$$

where $(Actual_{ij})$ represents the actual value for variable (j) in observation (i), (Mean (Actual)_j) is the mean of the actual values for variable (j), and $(Imputed_{ij})$ is the imputed value for variable (j) in observation (i).

5.4 Baseline methods

1) LWPMI [33] - Lightweight Window Portion-based Multiple Imputation (LWPMI) based on multivariate variables, correlation, data fusion, regression, and multiple imputations.
2) ImputeGAN [34] - A model based on generative adversarial networks (GANs) with an iterative strategy guided by complementary result gradients, to address missing values in multivariate time series data.
3) Time-Recurrent Variational AutoEncoder ODE (TRVAEODE) [35] - Integrating a time-aware LSTM encoder with neural ordinary differential equations (ODEs) to capture continuous-time latent dynamics for imputation and prediction of incomplete IoT time series data.
4) D-ANFIS [36]- A fuzzy inference system-based approach to handle missing data in Internet of Medical Things (IoMT) systems imputation by categorizing collected data into complete and incomplete datasets.

6. Type-style and fonts

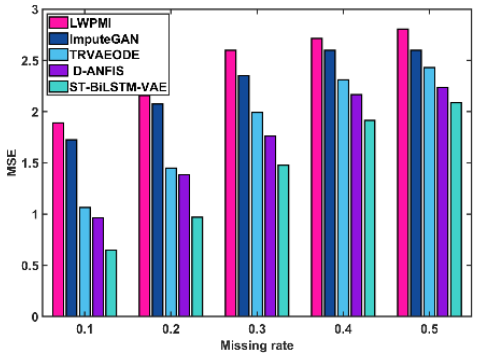
To evaluate the proposed method, two datasets IBRL and PRSA we used. The following subsections will provide the performance improvement of ST-Bi-LSTM-VAE over existing methods.

6.1 Evaluation with intel dataset

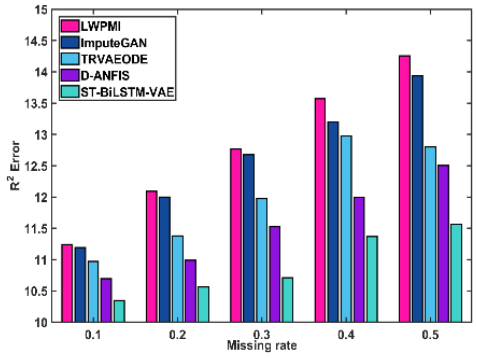
The performance of proposed method on IBRL dataset is depicted in Figure 5. The MSE values of remaining methods are high when compared with ST-Bi-LSTM-VAE. At missing gap of 10%, the MSE values obtained were 1.889, 1.724, 1.064, 0.963 and 0.649 for the methods LWPMI, ImputeGAN, TRVAEODE, D-ANFIS, and ST-BiLSTM-VAE respectively as shown in Figure .5(a). This is because LWPMI did not consider temporal correlations. ImputeGAN as this method was iterative method the performance of each iteration depends on previous iterations. D-ANFIS also not used spatial and temporal correlations.

The R^2 squared error for IBRL data at missing rate 20% were 12.09, 11.99, 11.37, 10.99, and 10.56. Similarly, R^2 squared error at 50% missing rate were 14.25, 13.93, 12.80, 12.50, and 11.560. Even the missing gap increased the proposed method had less increased in R^2 squared error value compared with existing state of art methods as shown in Figure. 5(b) because it utilized non-Euclidian spatial and temporal features effectively compared with existing methods. TRVAEODE had less performance compared with the ST-Bi-LSTM-VAE because of not utilising the spatial correlations and also TRVAEODE used Sine wave simulation for the purpose of training the model which may not accurately represent diverse real-world noise patterns.

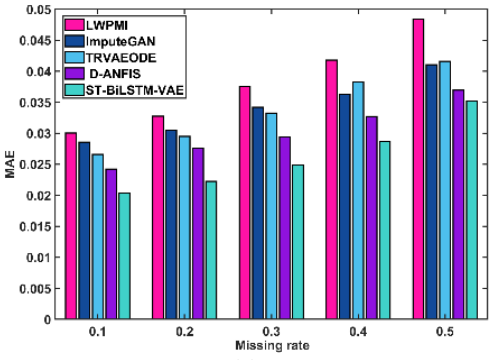
The MAE values of the existing methods were obtained high for exiting methods as shown in Figure 5(c). The MAE values of LWPMI, ImputeGAN, TRVAEODE, D-ANFIS, and ST-BiLSTM-VAE were 0.0300, 0.028, 0.0265, 0.0242 and 0.020 (at 10%) and 0.0327, 0.0304, 0.029, 0.0275 and 0.022 (at 20%). The decreased MAE values indicate the imputation accuracy of current methods. Considering the global spatial correlations and temporal correlations is the added advantage of ST-Bi-LSTM-VAE.



(a)

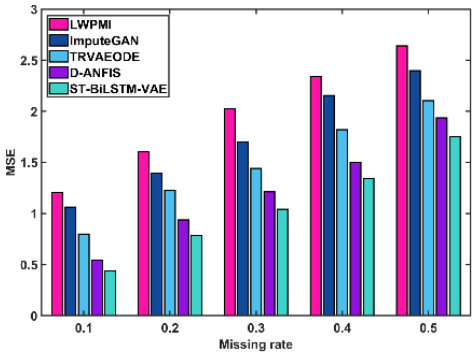


(b)

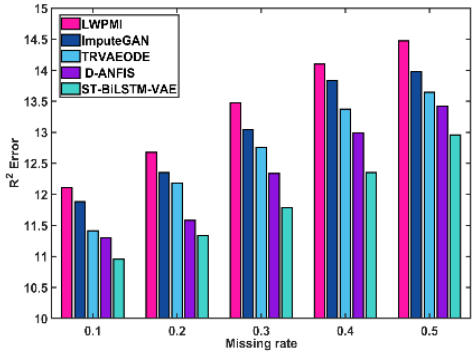


(c)

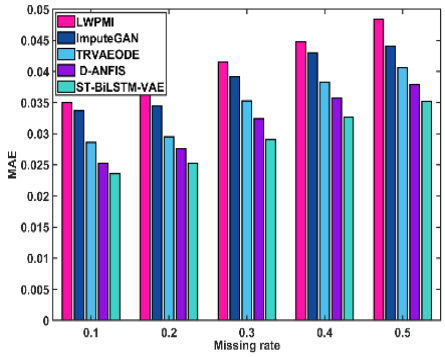
Figure. 5 Performance of IBRL dataset



(a)



(b)



(c)

Figure. 6 Performance of PRSA

6.2 Evaluation with PRSA dataset

Figure 6 shows the MSE, R^2 squared error and MAE values of baseline methods and ST-Bi-LSTM-VAE when working with PRSA dataset. At missing rate of 20%, the MSE values obtained for PRSA dataset were 1.60, 1.395, 1.225, 0.938, and 0.785988201. When MSE values of all the methods compared at 10% and 40% the differences were 1.13, 1.09, 1.02, 0.95, 0.90 illustrating that even the missing gap increases the increase of MSE value is less for the proposed method. The Bi-LSTM VAE used in ST-Bi-LSTM-VAE effectively finds the hidden correlations among the sensors data resulted in decreased imputation errors.

The R^2 squared error values for LWPMI, ImputeGAN, TRVAEODE, D-ANFIS, and ST-BiLSTM-VAE at missing gap 30% were 13.47, 13.044, 12.758, 12.33, and 11.78. Similarly, the MAE values at 10% missing were 0.035,0.033,0.028, 0.025, and 0.023. From Figure 6(a), 6(b) and 6(c), one can observe that ST-BiLSTM-VAE performed well when compared with existing methods. As the existing methods failed to utilise the spatial correlations such as ImputeGAN, TRVAEODE, D-ANFIS got more MAE, MSE and R^2 squared error. Though LWPMI taken the advantage of spatial correlations it requires more computational resources compared with ST-Bi-LSTM-VAE because of LGBM regression and data fusion. LWPMI also need high correlation among the data that is to be imputed.

The notable points about the ImputeGAN are that it was an iterative method, which may be complex to implement for dataset with large size. At the time of training ImputeGAN model all the missing gaps were filled with 0's led to missing of data pattern resulted in increased MSE, MAE and R^2 squared error.

TRVAEODE used sine wave simulation for traing the model. Simulating sine waves has limitations as it relies on fixed parameters such as frequency and amplitude, which may not fully capture real-world complexity. Adding Gaussian noise can improve realism, but it may not adequately represent the diverse noise patterns present in reality. Moreover, randomly initializing trajectories and selecting time points arbitrarily can introduce biases and overlook crucial details. Consequently, models trained on simulated data may face challenges when applied to real-world scenarios, underscoring the drawbacks of TRVAEODE methodology.

The D-Adaptive Network-based Fuzzy Inference System (D-ANFIS) combines artificial neural networks and fuzzy logic for function approximation. Despite its utility, ANFIS faced challenges such as complex parameter tuning, limited interpretability, sensitivity to initialization conditions, and difficulty in generalization. Also suffered from computational intensity, overfitting, and dependence on domain. Addressing these limitations requires careful parameter tuning, validation techniques, and consideration of model interpretability and generalization capabilities. Figure 5 and Figure 6 are evidents that the ST-Bi-LSTM-VAE model surpasses comparative models in the imputation task. Recognizing the challenges of long-distance dependency and gradient vanishing in traditional RNNs, the study opts for an improved variant, specifically Bi-LSTM, as a baseline. Notably, the ST-Bi-LSTM-VAE model outperforms various

baseline methods MDIMTFL, Impute GAN, TRVAEODE and D-ANFIS. The enhanced encoder of the ST-Bi-LSTM-VAE model effectively addresses the limitations of the GRU encoder, enabling it to learn the time intervals of time series and accurately infer the posterior distribution of irregularly-sampled time series data.

The Bi-LSTM encoder proves effective in learning the temporal distribution and GCN for spatial distribution, where the ST-Bi-LSTM-VAE model consistently outperforms other encoder methods across various missing rates.

7. Conclusion

To tackle the issue of multivariate missing data in IoT, our study introduced an innovative spatio-temporal imputation model that integrates Bi-LSTM, VAE and GCN. A series of comprehensive experiments were undertaken by creating significant missing intervals and varying missing rates between 10% to 50% across sensor-generated datasets featuring diverse characteristics IBRL (with more spatial and temporal correlations) and PRSA (with global spatial correlations) and evaluated with the measures MSE, MAE and R^2 squared error. The proposed method addressed non-Euclidian spatial dependencies within the network by incorporating GCN and a spatial Bi-LSTM. The augmentation of these correlations with temporal correlations helped to achieve better performance. This novel method worked better even there may be less Euclidian spatial correlations among the data which was proved when evaluated on PRSA dataset. ST-Bi-LSTM-VAE incorporated the temporal and global spatial correlations in the data so that it specifically suits for IoT data reconstruction compared with the existing methods. Furthermore, we enhance the robustness and stochastic generalization of our network by optimizing linear normalization flow and temporal concatenation for the latent variable of VAE. The effectiveness of our approach is demonstrated through experiments on two public datasets under various missing data scenarios, showcasing its feasibility compared to baseline methods. In future work, we aim to enhance the imputation accuracy by incorporating adversarial learning and leveraging auxiliary information. Furthermore, we plan to extend the application of our proposed model to the domain of traffic prediction with missing data.

Conflicts of Interest

The authors have no conflict of interests to declare that are relevant to the content of this article.

Author Contributions

Guggilam Venkata Vidyalakshmi contributed in conceptualization, methodology, software, data curation, formal analysis, writing—original draft. Gopikrishnan Sundaram contributed to conceptualization, methodology, supervision, result analysis, writing—review and editing final draft and validation.

References

[1] G. Shen, W. Zhou, W. Zhang, N. Liu, Z. Liu, and X. Kong, “Bidirectional spatial–temporal traffic data imputation via graph attention recurrent neural network”, *Neurocomputing*, Vol. 531, pp. 151–162, 2023.

[2] A. Gkillas and A. S. Lalos, “Missing data imputation for multivariate time series in industrial iot: A federated learning approach”, In: *Proc. of IEEE 20th International Conference on Industrial Informatics (INDIN)*, pp. 87–94, 2022.

[3] A. Feng and L. Tassiulas, “Adaptive graph spatial-temporal transformer network for traffic forecasting”, In: *Proc. of the 31st ACM International Conference on Information & Knowledge Management*, pp. 3933–3937, 2022.

[4] X. Liang, T. Zou, B. Guo, S. Li, H. Zhang, S. Zhang, H. Huang, and S. X. Chen, “Assessing beijing’s pm2.5 pollution: severity, weather impact, apec and winter heating”, In: *Proc. of the Royal Society A: Mathematical*, Physical and Engineering Sciences, Vol.471, 2015.

[5] S. Zhang, J. Chen, J. Chen, X. Chen, and H. Huang, “Data imputation in iot using spatio-temporal variational auto-encoder”, *Neurocomputing*, Vol. 529, pp. 23-32, 2023.

[6] W. Zhang, Y. Luo, Y. Zhang, and D. Srinivasan, “Solargan: Multivariate solar data imputation using generative adversarial network”, *IEEE Transactions on Sustainable Energy*, Vol. 12, No. 1, pp.743 – 746, 2021.

[7] H. Ahn, K. Sun, and K. P. Kim, “Comparison of missing data imputation methods in time series forecasting”, *Computers, Materials & Continua*, Vol. 70, No. 1, pp. 767–779, 2022.

[8] D. Adhikari, W. Jiang, J. Zhan, Z. He, D. B. Rawat, U. Aickelin, and H. A. Khorshidi, “A comprehensive survey on imputation of missing data in internet of things”, *ACM Computing Surveys*, Vol. 55, No. 7, pp 1–38, 2022.

[9] P. Bansal, P. Deshpande, and S. Sarawagi, “Missing value imputation on multidimensional

time series", *arXiv preprint*, arXiv:2103.01600, 2021. https://doi.org/10.48550/arXiv.2103.01600.

[10] G. Folino, C. Otranto Godano, and F.S. Pisani, "An ensemble-based framework for user behaviour anomaly detection and classification for cybersecurity", *Journal of Supercomputing*, Vol. 79, pp. 11660–11683, 2023.

[11] P. S. Raja, K. Sasirekha, and K. Thangavel, "A Novel Fuzzy Rough Clustering Parameter-based missing value imputation", *Neural Computing and Applications*, Vol. 32, pp 10033–10050, 2020.

[12] S. C.X. Li, B. Jiang, and B. Marlin, "Misgan: Learning from incomplete data with generative adversarial networks", *arXiv preprint arXiv:1902.09599*, 2019. https://doi.org/10.48550/arXiv.1902.09599.

[13] J. Yoon, J. Jordon, and M. Schaar, "Gain: Missing data imputation using generative adversarial nets", In: *Proc. of International conference on machine learning, PMLR*, pp. 5689–5698, 2018.

[14] Y. Luo, Y. Zhang, X. Cai, and X. Yuan, "E2gan: End-to-end generative adversarial network for multivariate time series imputation", In: *Proc. of the 28th international joint conference on artificial intelligence*, AAAI Press Palo Alto, CA, USA, pp. 3094–3100, 2019.

[15] Q. Ma, S. Li, and G. W. Cottrell, "Adversarial joint-learning recurrent neural network for incomplete time series classification", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol 44, No. 4, pp.1765–1776, 2020.

[16] X. Zhou, Y. Li, and W. Liang, "Cnn-rnn based intelligent recommendation for online medical pre-diagnosis support", *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, Vol. 18, No. 3, pp.912 – 921, 2021.

[17] M. Hou, T. Tang, F. Xia, I. Sultan, R. Kaur, and X. Kong, "MissII: Missing Information Imputation for Traffic Data", *IEEE Transactions on Emerging Topics in Computing*, pp.1-14, 2023.

[18] L. Chen, D. Chen, Z. Shang, B. Wu, C. Zheng, B. Wen, and W. Zhang, "Multi-Scale Adaptive Graph Neural Network for Multivariate Time Series Forecasting", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 35, No. 10, pp. 10748-10761, 2023.

[19] JZ. Chen, ZK. Lü, and HM. Lin, "Prediction Model for Traffic Flow with Missing Values Based on Generative Adversarial and Graph Convolutional Networks", *Journal of Highway and Transportation Research and Development*, Vol.17, No.3, pp. 62-74, 2023.

[20] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Stacked bidirectional and unidirectional lstm recurrent neural network for forecasting network-wide traffic state with missing values", *Transportation Research Part C: Emerging Technologies*, Vol. 118, pp.102674, 2020.

[21] 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 of Things*, Vol. 6, No. 4, pp. 6618 – 6628, 2019.

[22] X. Song, J. Li, Q. Lei, W. Zhao, Y. Chen, and A. Mian, "Bi-clkt: Bi-graph contrastive learning based knowledge tracing", *Knowledge-Based Systems*, Vol. 241, pp 108274, 2022.

[23] Z. Zhang, X. Lin, M. Li, and Y. Wang, "A customized deep learning approach to integrate network-scale online traffic data imputation and prediction", *Transportation Research Part C: Emerging Technologies*, Vol. 132, 2021.

[24] 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, 2021.

[25] D. Xu, H. Peng, C. Wei, X. Shang, and H. Li, "Traffic state data imputation: An efficient generating method based on the graph aggregator", *IEEE Transactions on Intelligent Transportation Systems*, Vol. 23, No. 8, pp.13084 – 13093, 2022.

[26] W. Zhang, H. Wang, and F. Zhang, "Spatio-temporal Fourier enhanced heterogeneous graph learning for traffic forecasting", *Expert Systems with Applications*, Vol. 241, pp. 122766, 2024.

[27] D. Xu, H. Peng, Y. Tang, and H. Guo, "Hierarchical spatio-temporal graph convolutional neural networks for traffic data imputation", *Information Fusion*, Vol. 106,2024.

[28] 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.

[29] M. Canizo, I. Triguero, A. Conde, and E. Onieva, "Multi-head cnn–rnn for multi-time series anomaly detection: An industrial case

study", *Neurocomputing*, Vol. 363, pp. 246-260, 2019.

[30] Q. Zhou, S. He, H. Liu, J. Chen, and W. Meng, "Label-free multivariate time series anomaly detection", *IEEE Transactions on Knowledge and Data Engineering*, pp. 1 – 15, 2024.

[31] G. Zheng, Y. Yang, and J. Carbonell, "Convolutional normalizing flows", *arXiv preprint arXiv:1711.02255*, 2017.

[32] D. Adhikari, W. Jiang, J. Zhan, M. Assefa, H. A. Khorshidi, U. Aickelin, and D. B. Rawat, "A lightweight window portion-based multiple imputation for extreme missing gaps in iot systems", *IEEE Internet of Things Journal*, Vol 11, issue 3, pp. 3676 – 3689, 2024.

[33] R. Qin and Y. Wang, "Imputegan: Generative adversarial network for multivariate time series imputation", *Entropy*, Vol.25, No. 1 pp:137, 2023.

[34] Z. Chang, S. Liu, R. Qiu, S. Song, Z. Cai, and G. Tu, "Time-aware neural ordinary differential equations for incomplete time series modelling", *The Journal of Supercomputing*, Vol. 79, No.16, pp.18699-18727, 2023.

[35] H. Turabieh, M. Mafarja, and S. Mirjalili, "Dynamic adaptive network-based fuzzy inference system (d-anfis) for the imputation of missing data for internet of medical things applications", *IEEE Internet of Things Journal*, Vol. 6, No.6, pp. 9316 – 9325, 2019.