Hindawi
Complexity
Volume 2021, Article ID 3320402, 12 pages
https://doi.org/10.1155/2021/3320402

Research Article

RAEF: An Imputation Framework Based on a Gated Regulator Autoencoder for Incomplete IIoT Time-Series Data

Jianyong Zhao,[1] Jiachen Qiu,[1] Danfeng Sun,[2] and Baiping Chen[1]

[1]*Institute of Intelligent and Software Technology, Hangzhou Dianzi University, Hangzhou 310018, China*
[2]*Institut f. Automation und Kommunikation, Magdeburg 39106, Germany*

Correspondence should be addressed to Baiping Chen; chenbp@hdu.edu.cn

Received 23 August 2021; Accepted 29 October 2021; Published 8 December 2021

Academic Editor: Fanlin Meng

The number of intelligent applications available for IIoT environments is growing, but when the time-series data these applications rely on are incomplete, their performance suffers. Unfortunately, incomplete data are all too frequent to a phenomenon in the world of IIoT. A common workaround is to use imputation. However, the current methods are largely designed to reconstruct a single missing pattern, where a robust and flexible imputation framework would be able to handle many different missing patterns. Hence, the framework presented in this study, RAEF, is capable of processing multiple missing patterns. Based on a recurrent autoencoder, RAEF houses a novel neuron structure, called a gated regulator, which reduces the negative impact of different missing patterns. In a comparison of the state-of-the-art time-series imputation frameworks at a range of different missing rates, RAEF yielded fewer errors than all its counterparts.

1. Introduction

Today's IIoT sensors are capable of collecting an inordinate amount of data, and the applications built to process these data are allowing us to monitor, analyze, and understand how things in our physical world are changing over time [1]. However, to continue improving our capacity for time-series analysis, it is not enough to improve just the analysis methods with better context recognition [2], expanded service recommendations [3], improved anomaly detection [4], and so on. The quantity and quality of the time-series data also need to be improved. For the most part, improving data quality means making sure a data stream is comprehensive and complete. Scope tends to be the easier of these two to address—simply adding more and different types of sensors will get the job done. Unfortunately, completeness is often a more common and difficult issue to overcome [5]. Data can be incomplete due to noise, sensor malfunctions, equipment error, human error, incorrect measurements, and other unavoidable circumstances [6]. As such, almost every data stream produced by a sensor will be incomplete to some degree [7].

Being so common, there are several methods of dealing with incomplete data. The first is to install redundant sensors as backups. If one sensor fails to capture some data, the other may not. The main drawback with this solution is that two sensors cannot be in exactly the same place, nor do they tend to operate on exactly the same timing, so it can be difficult to align the temporal and spatial characteristics of the data [8]. Hence, a more common remedy has been some form of data manipulation: generally, either deletion or imputation [9].

Deletion is a simple and efficient answer when the amount of missing data is very small in comparison with the total. However, in applications that are very sensitive to time series, deleting a small number of records can be enough to destroy the coherence of a sequence and may seriously affect the correctness of the results. Further, most data analysis methods, especially machine-learning methods, require a complete set of time stamps and are not robust to missing data. In contrast, imputing missing data can reduce sensitivity and provide a complete set of time stamps. Hence, imputation has commanded the bulk of the research focus in recent decades [10, 11].

The easiest methods of imputation simply replace the missing information with statistically reasonable values, such as means, modes, medians, or any predefined value [12]. However, while straightforward and convenient, the accuracy of such methods relies on the complexity of the samples. With small, basic samples, they work fine, but when the features become complex, these methods are not reliable. For example, imputing with multivariate data generally requires an algorithm based on clustering. Similar samples are grouped into the same cluster and then used to evaluate missing values group-by-group [13]. Rahman and Islam [14] rough fuzzy k-means algorithm is one example of this type of clustering imputation. Here, the researchers exploited fuzzy expectation-maximization and fuzzy clustering to build a missing value imputation framework for data preprocessing. Raja and Sasirekha [15] method was designed to handle missing values, while Zhao et al. [8] developed a local similarity imputation method that estimates missing data based on the stacked autoencoder (SAE) fast clustering algorithm and the top k-nearest neighbors. There is no doubt that these clustering-based imputations methods yield excellent results. However, clustering an entire time series is hugely time-consuming to the extent that these approaches cannot keep pace with today's dramatic increases in data volume. Additionally, there comes a point when the data may be too incomplete for these methods to work with any level of accuracy.

In the IIoT paradigm, sensor data have special properties. For instance, multiple sensors are often used to record the same/similar measurements in many systems [16]. Sensors that are geographically close to each other tend to be highly correlated for certain periods of time [17]. This means that missing data can sometimes be imputed from the associated sensors, whether spatially or temporally. In these situations, modeling time series and then applying an imputation method such as smoothing or interpolation [18] can be a good choice.

Generally, smoothing or interpolation methods have a low computational overhead and are simple to implement, although they are not suitable for finding long-term correlations in time-series data. Machine-learning techniques can correlate features, which can improve imputation performance, such as generative adversarial models [19–22] and recurrent neural networks (RNNs). Among them, RNNs are known to be good at modeling time series, and for this reason, many hybrid-RNN methods have been developed. This is because vanilla RNNs estimate missing values from the data immediately preceding the gap. For instance, Kim et al. [23] devised an RNN model to impute missing medical examination data. The time series are modeled by RNNs, which compensate for the missing measurements and then predict future values. Minseok et al. [24], for example, developed an imputation framework called DeepIN based on this type of correlation information. DeepIN uses a deep network consisting of multiple LSTMs arranged according to the correlation information of each IIoT device. Ma et al.'s [25] LIME-RNN models incomplete time series (linear memory vector recurrent neural network). A learnable linear combination of previous history states means gradient information can be propagated efficiently. In this way, LIME-RNN can take full advantage of the previously observed information to reduce the negative impact of missing values. Alternatively, Li et al. [26] proposed a multi-view learning method for estimating missing values in time-series traffic data that combine RNNs and collaborative filtering techniques. There is a large body of papers on imputing with incomplete time series that assume any missing data from the current time step are the same as the previous time step [25, 27, 28] or that apply a decay mechanism to a hidden state to impute the missing data [29–31]. Yet, with RNNs, imputation performance suffers when the missing values become continuous. Further, the above imputation strategies can lead to instability during training, and with high missing rates, decay mechanisms will not find sufficient hidden information.

Another branch of investigation in the search to improve imputation performance is missing patterns. In this stream, Minseok et al. [24] compared the effects of missing continuity and discontinuity on imputation performance. Anindita et al. [32] and Tsai and Chang [33] considered the missing patterns of arbitrariness and monotonicity in medical data. Insuwan et al. [34] found that the rating data present a special missing pattern caused by user preference genres. Tak et al. [35] distinguished and contrasted the missing patterns in traffic data caused by prolonged physical damage to the sensors and measurement noise. However, to the best of our knowledge, no special missing patterns for IIoT environments have been proposed. This study is an attempt to change that. As such, our contributions are as follows:

(1) We propose a framework based on a recurrent autoencoder, called RAEF. The encoder turns an incomplete time series into vector representations of both local information and global information. The decoder then initializes using the global information, decoding the local information into complete time-series data.

(2) As an alternative to decaying the hidden state, inside RAEF, a gated regulator focuses on discriminating between ground truth information and fictitious information. This mechanism is better able to reduce the negative impact of increased missing rates in different missing patterns.

(3) In empirical evaluations in a real IIoT environment, RAEF proves to be effective. Additionally, comparisons between RAEF and several state-of-the-art frameworks demonstrate that RAEF results in fewer errors at each missing rate tested.

The remainder of this study is organized as follows. Section II. presents the problem formulation and some necessary preliminaries. Section III. describes RAEF's structure. Sections II. and IV. present the details of the experiments and results, and Section V. concludes the study.

2. Preliminary

2.1. Incomplete Time-Series Data. A sequential time-series data $\mathbf{X} = \{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^T\}$ are a sequence of T observations. At each time step t, the observation $x^t \in \mathbb{R}^D$ has D features

$\{x_1^t, x_2^t, \ldots, x_D^t\}$. A sequential binary missing mask $\mathbf{M} = \{\mathbf{m}^1, \mathbf{m}^2, \ldots, \mathbf{m}^T\}$, $\mathbf{m}^t \in \{0,1\}^D$ is applied when generating representations of the data, where m^t denotes which features are missing at time step t. The features missing at time step t can be described as follows:

$$m_D^t = \begin{cases} 0, & \text{if } x_t^d \text{ is missing,} \\ 1, & \text{otherwise.} \end{cases} \tag{1}$$

Thus, an incomplete sequential time series is denoted as $\overline{\mathbf{X}} = \{\overline{\mathbf{x}}^1, \overline{\mathbf{x}}^2, \ldots, \overline{\mathbf{x}}^T\}$, $\overline{\mathbf{x}}^t \in \mathbb{R}^D$.

The following rule is applied when training the model to create an artificial incomplete time series:

$$\overline{\mathbf{x}}^t = \mathbf{x}^t \odot \mathbf{m}^t. \tag{2}$$

2.2. Analysis of Missing Pattern. With an analysis of a large amount of time-series data from the real IIoT environment, a piece of knowledge is that the main missing pattern for ITS is two types: univariate missing and common-mode missing.

Univariate missing data are the most common pattern, which often appears as a series of reading losses in a single sensor over a short period of time, as shown in Figure 1. The usual cause is a fault in the sensor itself. For simplicity, we have only considered recoverable cases in this study—namely where the data collection can be recovered in a limited time. Here, $k_{\max}$ is the maximum length of continuous missing data, noting that, in general, $k_{\max} = D$.

The other type of missing pattern is common-mode missing data, also known as common-mode failure. In these cases, a large number of sensors fail to upload their readings at the same time. Usually, this is caused by some external factor, such as a disk error, a network communications error, and human intervention. [36]. Figure 2 shows an example of this type of missing pattern.

2.3. Recurrent Neural Networks. Recurrent neural networks (RNNs) are especially suited to dealing with temporally and spatially correlated information because they process historical information recursively and model historical memory. RNNs are neural networks that work on a variable length sequence $\mathbf{X} = \{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^T\}$ by maintaining a hidden state $\mathbf{h}$ over time. At each time step t, the hidden state $\mathbf{h}^t$ is updated by the following equation:

$$\begin{cases} \mathbf{z}^t = \mathbf{W}_i \mathbf{x}^t + \mathbf{W}_h \mathbf{h}^{t-1} + \mathbf{b}_h, \\ \mathbf{h}^t = f\left(\mathbf{z}^t\right), \\ \mathbf{y}^t = f\left(\mathbf{W}_o \mathbf{h}^t + \mathbf{b}_o\right), \end{cases} \tag{3}$$

where f is an activation function. Often f is as simple as performing a linear transformation on the input vectors, summing them, and applying an element-wise logistic sigmoid function. z_t is an internal intermediate state, and the model parameters are symbolized by $\mathbf{W}_i$, $\mathbf{W}_h$, $\mathbf{W}_o$, $\mathbf{b}_o$, and $\mathbf{b}_h$. Further, we can simplify the RNN at time step t as an $\mathcal{F}_{\text{RNN}}$ function formulated by the following equation:

$$\mathbf{h}^t = f_{\text{RNN}}\left(\mathbf{h}^{t-1}, \mathbf{x}^t\right), \tag{4}$$

where f_{RNN} encapsulates the different RNN variants. LSTMs [37] and gated recurrent units (GRUs) [38] are both very popular RNN variants.

3. The RAEF Imputation Framework

Figure 3 shows the structure of the RAEF. It learns to encode a sequence that may contain missing data and then to decode those vectors back into sequential time-series data without missing data. Note that the basic neuron used in the RAEF includes a novel GR.

3.1. RNN Encoder. The encoder is a model based on an RNN or variant. In our case, since $\mathbf{x}^t$ may have missing data, it cannot be used to update $\mathbf{h}^t$ as per Equation (4). So, when $\mathbf{x}^t$ is missing, the output of the previous time step is used instead. The information in this previous time step is a type of local information. Further, the mean of $\mathbf{x}^t$ across time steps, denoted as $\mathbf{r}^t$ and $\widetilde{\mathbf{x}}$. $\widetilde{\mathbf{x}}$, can be described as follows:

$$\widetilde{\mathbf{x}} = \frac{\sum_{t=1}^{T}\left(\mathbf{x}^t \odot \mathbf{m}^t\right)}{\sum_{t=1}^{T}\left(\mathbf{m}^t\right)}. \tag{5}$$

Formally, the initial hidden state h_0 is initialized as an all-zero vector. From $t = 1$ to T, the model is updated by the following equation:

$$\begin{cases} \mathbf{r}^t = \mathbf{W}_r \mathbf{h}^{t-1} + \mathbf{b}_r, \\ \widehat{\mathbf{x}}^t = (1-\gamma)\mathbf{r}^t + \gamma \widetilde{\mathbf{x}}, \\ \dot{\mathbf{x}}^t = \left(1 - \mathbf{m}^t\right) \odot \widehat{\mathbf{x}}^t + \mathbf{m}^t \odot \mathbf{x}^t, \\ \mathbf{h}^t = f_{\text{RNN}}\left(\mathbf{h}^{t-1}, \dot{\mathbf{x}}^t\right), \end{cases} \tag{6}$$

where γ is a learnable scalar, initialized as 0. Introducing a learnable γ allows the network to first rely on the cues in $\widetilde{\mathbf{x}}$. Gradually, it learns to assign more weight to $\mathbf{r}^t$. Hence, the encoder can be described as $E(x, h_e; \theta_e)$, where x is the sequential input, h_e is a hidden state, and E is a differentiable function represented by Equation (6) with the parameters θ_e. Once the sequential time-series data $\mathbf{X}$ have been fed into the encoder, $\mathbf{R} = \{\mathbf{r}^1, \mathbf{r}^2, \ldots, \mathbf{r}^T\}$ is recorded, and a vector $\mathbf{h}_c$ is generated that contains global information about the full sequence of time-series data input:

$$\mathbf{h}_c = g\left(\left\{\mathbf{h}^1, \mathbf{h}^2, \ldots, \mathbf{h}^T\right\}\right), \tag{7}$$

where $g(.)$ are some nonlinear functions. Here, we consider a simple deployment and so assume that $g(\{\mathbf{h}^1, \mathbf{h}^2, \ldots, \mathbf{h}^T\}) = \mathbf{h}^T$. The loss function of the encoder is as follows:

$$\begin{aligned} &\ell_{\text{encoder}} = \frac{1}{T-1}\sum_{t=1}^{T-1} \lambda_{t+1} \mathscr{L}_e\left(\mathbf{r}^{t+1}, \mathbf{x}^{t+1}\right), \\ &\text{s.t.} \sum_{t=1}^{T-1} \lambda_{t+1} = 1, \end{aligned} \tag{8}$$

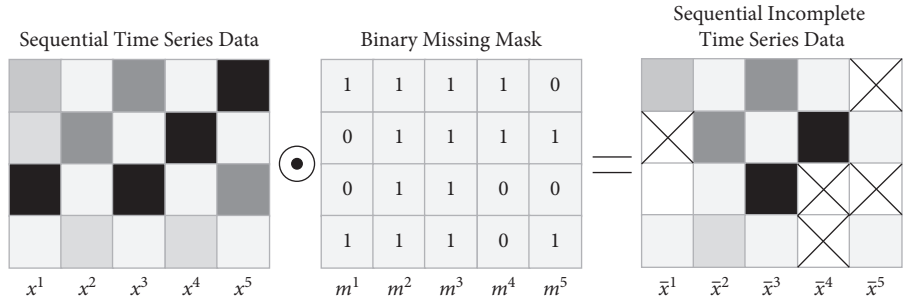

Figure 1: An example of a univariate missing data pattern.

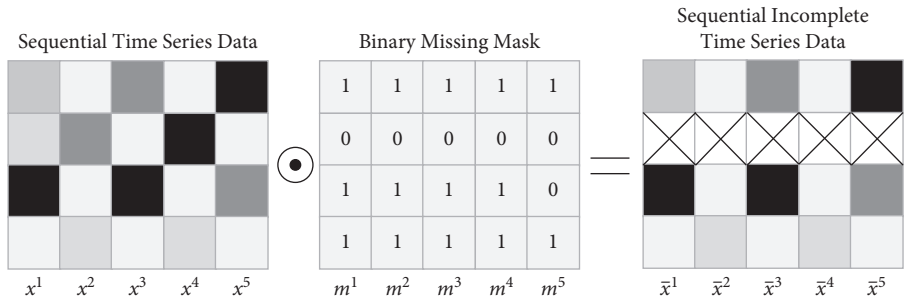

Figure 2: An example of a common-mode failure pattern.

where λ_t is a coefficient weighting, which represents the importance of the previous imputation at each time step. Intuitively, it does not need to be overly precise for the first few time steps of training the encoder. In common, it is assumed that:

$$\lambda_{t+1} = \frac{t+1}{\sum_{T=1}^{t-1}(t+1)}. \tag{9}$$

3.2. RNN Decoder. The decoder is also a model based on an RNN or variant that aims to decode the sequence R from the encoder back into a sequential time-series data without missing data. c is used to initialize the hidden state of the decoder. Note that, according to Equation (6), $\mathbf{R} = \{\mathbf{r}^1, \mathbf{r}^2, \ldots, \mathbf{r}^T\}$ is considered to be a replacement to $\{\mathbf{x}^2, \mathbf{x}^3, \ldots, \mathbf{x}^{T+1}\}$. Hence, the decoder works backwards, reading the sequence in the reverse order (i.e., from $\mathbf{r}^T$ to $\mathbf{r}^1$). The sequential outputs of the decoder can be derived using Equation (3), denoted as $\{\mathbf{y}^T, \mathbf{y}^{T-1}, \ldots, \mathbf{y}^1\}$.

$$\begin{cases} \mathbf{h}^t = \mathscr{F}_{de}\left(\mathbf{h}^{t-1}, \mathbf{r}^t\right), \\ \mathbf{y}^t = \mathbf{W}_y \mathbf{h}^{t-1} + \mathbf{b}_y. \end{cases} \tag{10}$$

Hence, the decoder can be described as $D(r, h_d; \theta_d)$. Finally, the decoder trains the parameters by minimizing the errors between the output y^t and the input sequential time-series data $\mathbf{x}^t$. The loss function is defined as follows:

$$\ell_{\text{decoder}} = \frac{1}{T}\sum_{t=1}^{T}\sum_{d=1}^{D}\left(1 - m_D^t\right)\mathscr{L}_e\left(y_D^t, x_D^t\right), \tag{11}$$

where $\mathscr{L}_e$ uses the absolute error,

$$\mathscr{L}_e(x, y) = |x - y|. \tag{12}$$

3.3. Gated Regulator. Since the operation of the encoder is represented in Equations (5) and (6), the input data of each time step are not completely consistent in authenticity. Intuitively, if the imputation framework can evaluate the input data authentically at an early stage, and before calculating the candidate state, the hidden state can reduce the incidence of inaccurate information. A gated structure, i.e., a gated regulator, is therefore integrated into the encoder, as shown in Figure 4. The motivation is to allow the encoder to decide how much of the current hidden state $\mathbf{h}^t$ will gain its information from the current input without increasing the extra information. Formally, this can be described as follows:

$$\mathbf{g}^t = \sigma\left(\mathbf{W}_g \cdot \left[\mathbf{m}^t, \mathbf{h}^{t-1}\right]\right). \tag{13}$$

Equation (6) becomes

$$\begin{cases} \mathbf{r}^t = \mathbf{W}_r \mathbf{h}^{t-1} + \mathbf{b}_r, \\ \tilde{\mathbf{x}}^t = (1-\gamma)\mathbf{r}^t + \gamma\tilde{\mathbf{x}}, \\ \dot{\mathbf{x}}^t = \left(1 - \mathbf{m}^t\right) \odot \tilde{\mathbf{x}}^t + \mathbf{m}^t \odot \mathbf{x}^t, \\ \mathbf{g}^t = \sigma\left(\mathbf{W}_g \cdot \left[\mathbf{m}^t, \mathbf{h}^{t-1}\right]\right), \\ \tilde{\mathbf{h}}^t = f_{RNN}\left(\mathbf{h}^{t-1}, \dot{\mathbf{x}}^t\right), \\ \mathbf{h}^t = \left(1 - g^t\right) \odot \tilde{\mathbf{h}}^t + \mathbf{e}^t \odot \mathbf{h}^{t-1}. \end{cases} \tag{14}$$

Note that the gated regulator is an independent structure, which means it must be compatible with the RNN or variants. As an example, LSTM-GR means an LSTM with the gated regulator.

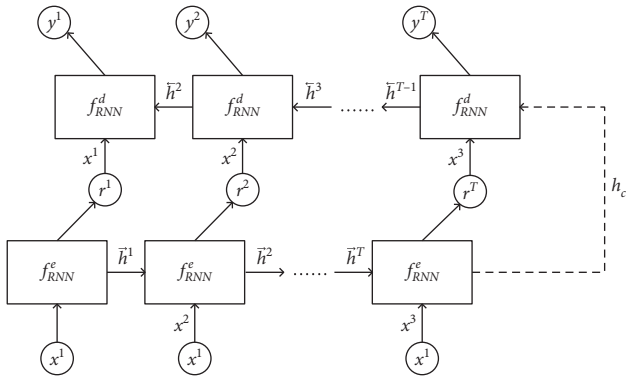

Figure 3: Proposed RAEF structure.

3.4. Training Process. To prevent a vanishing gradient or problems with explosion while back propagating RAEF, the training algorithm, Algorithm 1, prescribes that the encoder and decoder are trained asynchronously. The training process is therefore divided into three parts:

(1) Input x into the encoder, and update the encoder by descending its gradient $\nabla_{\theta_e}\ell_{\text{encoder}}$.

(2) Record the encoder's output r

(3) Input r into the decoder, and update the decoder by descending its gradient $\nabla_{\theta_d}\ell_{\text{decoder}}$.

Throughout, weight clipping is used to limit changes in the encoder's gradient.

4. Experiments Details

Our experience of real-world IIoT data, as shown in Figure 5, is that a great many data points can be missing from time series collected in these environments. The levels shown in Figure 5 indicate just how widespread the problem of missing data is in IIoT environments. This disturbing phenomenon not only affects the ability to monitor devices in real time but also reduces the accuracy of any subsequent analysis done by downstream applications.

In a series of analyses, we compare imputation with RAEF to several state-of-the-art imputation frameworks based on RNNs. Then, we illustrated how incomplete time-series imputation can improve the effectiveness of data applications. Last, we discuss the choice of T.

4.1. Dataset and Experiment Setup. The datasets used in the experiment are summarized in Table 1.

4.1.1. UCI Air Quality Data (UAQ). The UCI dataset contains 9358 records of average hourly responses from an array of 5 metal oxide chemical sensors embedded in an air quality chemical multi-sensor device taken between March 2004 and February 2005. The air quality data points have 12 features, and 7.5% of the values are missing. After removing the records with missing data, we randomly selected 20% of the data for testing and the others for training. Pearson's correlations between each feature are shown in Figure 6. This dataset can be thought of as an incomplete time-series dataset of a real IIoT environment that is rich in information and has a low- to middle-level missing rate.

4.1.2. Base Station Status Data (BSS). This dataset was collected from an ePLCM002FR edge node, developed by Hangzhou Yiyitaidi Information Technology Co., Ltd. and deployed in a base station located at the Spring Shopping Mall in the Zhangdian District, Zibo, Shandong Province (see Figure 7). The dataset comprises 14,820 data readings taken between February 2018 and February 2019. Every data point contains six attributes: the temperature and current intensity of two rectifiers, the air conditioning setting temperature, and environmental temperature. 18.2% of the values are missing. We used the data collected for May and September 2018 and February 2019 for testing. The remaining data were used for training.

Pearson's correlations between each feature are shown in Figure 8. Compared with the UAQ dataset, the BSS dataset has shorter collection cycles, low data dimensions, and a higher missing rate. To stabilize the training with each dataset, we normalized the raw data via a linear transformation using the maximum and minimum (min-max normalization) before the experiment. However, because the BSS dataset does not contain any ground truth labels, experimenting with the actual missing values was not possible. Thus, we simulated missing data by randomly omitting data according to different missing rates, and using the real values as a ground truth label in Table 1 provides the details.

The results were assessed in terms of mean absolute error (MAE) and mean relative error (MRE), calculated as follows:

$$\begin{aligned}\text{MAE} &= \frac{\sum_{i\in\Omega}\left|x_i - \widehat{x}_i\right|}{S(\Omega)},\\ \text{MRE} &= \frac{\sum_{i\in\Omega}\left|x_i - \widehat{x}_i\right|}{\sum_{i\in\Omega}\left|x_i\right|},\end{aligned} \tag{15}$$

where Ω denotes the index set of the missing values, and $S(.)$ denotes the size. x_i is the ground truth of the ith missing item, and $\widehat{x}_i$ is its imputed value.

4.2. Baselines. Brief descriptions of the comparators we chose as baselines are as follows:

(1) Border mean (BM)—uses the average of the previous and posterior record of the missing values as the imputed value.

(2) K-Nearest neighbor (KNN)—uses KNN [40] with a fixed $k = 6$ to find similar samples and imputes the missing values according to the weighted average of the neighbors.

(3) BRITS [29]—a novel method based on RNN that combines a bidirectional recurrent hidden state decay mechanism and forward imputation. This approach can impute the missing values in a

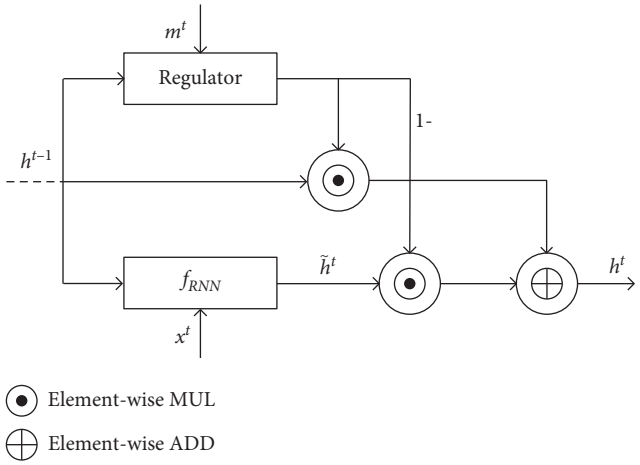

Figure 4: Simple structure of a gated regulator.

bidirectional recurrent dynamical system without any specific assumptions.

(4) LIME-LSTM [25]—a novel framework for modeling incomplete time series based on LIME-RNN using an LSTM, where a network learns the residual connection between time steps and implements a linear combination of previous historical states.

Note that BM and KNN are common approaches to imputation. BRITS and LIME-LSTM are both imputation frameworks for time series based on RNNs.

4.3. Implementation Details. We developed two implementations of RAEF: one with an LSTM and the other with a GRU. Further, we configured each model with and without a gated regulator to result in four baselines as follows.

For the proposed RAEF implementation, we tried two RNN variants: LSTM and GRU. Hence, we implemented 4 kinds of RAEF, RAEF-LSTM, RAEF-GRU, RAEF-LSTM-GR, and RAEF-GRU-GR:

(1) RAEF-LSTM (RFL): both the encoder and decoder are implemented as LSTMs

(2) RAEF-LSTM-GR (RFLR): the encoder was an LSTM incorporating a gated regulator, and the decoder was an LSTM

(3) RAEF-GRU (RFG): both the encoder and decoder were implemented as GRUs

(4) RAEF-GRU-GR (RFGR): the encoder was implemented as a GRU with a gated regulator, and the decoder was a GRU.

For all methods, we fixed the parameters of the RNNs to be the same. The dimensions of the hidden state were $n = 64$, and the learning rate was $\alpha = 0.0001$. In deploying the RNN-based models, we cut the datasets into sequences with a fixed length of T and input ms samples at once for training. The settings for the values of T and ms are shown in the last two lines of Table 1 and were applied to all RNNs consistently. Note that, in the training process, instead of using a validation set, we ended the training when the training loss leveled off.

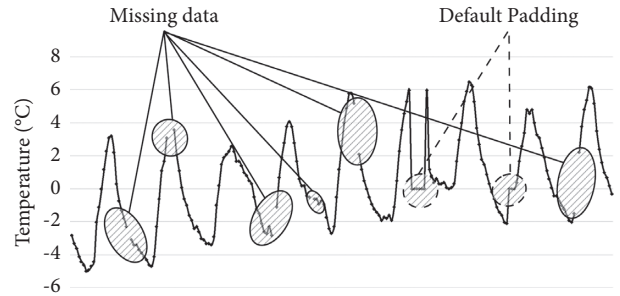

Figure 5: Change chart of temperature sensor readings of a base station in a large cellular network [39], collected from January 1 to 10, 2019. Around 20% of the data points in the time series are missing.

Our experimental procedure had three main steps. First, we randomly deleted data from the complete time series to mimic the different missing patterns and with different missing rates. We then split the data into training and testing sets according to the proportions mentioned in Section A. Second, we trained all the frameworks. Third, we used different frameworks to generate imputation results for the testing set and evaluated the frameworks by comparing results with the ground truth data in terms of the evaluation metrics.

All experiments were run on the TensorFlow platform using an Intel Core i7-8700K, 3.60-GHz CPU with 16-GB RAM, and a GeForce RTX 2080 8G.

5. Result and Discussion

5.1. Imputation Performance for Single Missing Pattern. Tables 2 and 3 show the results of the imputations, where MP denotes missing pattern and MR stands for missing rate. From these results, we drew the following observations.

(1) Border mean (BM) was quite inaccurate and became less accurate as the missing rate increased.

(2) KNN was not effective for imputing missing values with the common-mode pattern because the distance between the samples often could not be measured given the complete loss of all attributes. KNN was able to achieve a result with the univariate-type missing patterns at low missing rates but was sensitive to changes in this rate, and its performance grew worse as the rate increased.

(3) LIME-LSTM, with its unidirectional RNNs, did not perform as well the frameworks that contain a bidirectional RNN, i.e., BRITS and RAEF.

(4) LIME-LSTM and BSS could not cope with high missing rates. At low missing rates, RAEF and BRITS demonstrated similar performance. However, as the missing rate increases, RAEF performed significantly better than BRITS, especially the LSTM-GR implementation.

Input: The learning rate α. The clipping parameter c. The batch size ms. The set of training data X. The initial encoder parameters θ_e. The initial decoder parameters θ_d.

while not converged **do**
  Sample $\{x^{(i)}\}_{i=1}^{ms}$ from X.
  Set $\{m^{(i)}\}_{i=1}^{ms}$ according to the missing pattern.
  Initialize h_e by zeros.
  for $i = 1$ to ms **do**
    $r^{(i)} \leftarrow E(x^{(i)}, h_e; \theta_e)$.
    $g_{\theta_e} \leftarrow g_{\theta_e} + \nabla_{\theta_e} \ell_{\text{encoder}}$.
  end for
  $\theta_e \leftarrow \theta_e + \alpha \cdot \text{RMSProp}\ (\theta_e, g_{\theta_e}/ms)$.
  $\theta_e \leftarrow \text{clip}\ (\theta_e, -c, c)$.
  $\{r^{(i)}\}_{i=1}^{ms} \leftarrow \{E(x^{(i)}, h_d; \theta_e)\}_{i=1}^{ms}$.
  Get $\{h_c^{(i)}\}_{i=1}^{ms}$ by Equation (7).
  for $i = 1$ to ms **do**
    $h_d \leftarrow h_c^{(i)}$
    $y^{(i)} \leftarrow D(r^{(i)}, h_d; \theta_d)$.
    $g_{\theta_d} \leftarrow g_{\theta_d} + \nabla_{\theta_e} \ell_{\text{decoder}}$.
  end for
  $\theta_d \leftarrow \theta_d + \alpha\ \text{RMSProp}\ (\theta_d, g_{\theta_d}/ms)$.
end while

Algorithm 1: Training of RAEF.

Table 1: Details of experimental dataset.

Name	UCI air quality data	Base station status data
Records length	9358	14 820
Dimensions	12	7
Missing rate	7.5%	18.2%
Collection cycle	Per 1 h	Per 30 min
T	18	14
ms	200	1000

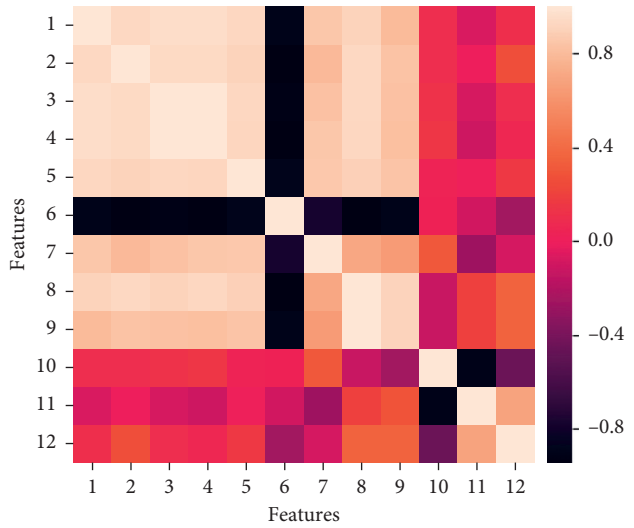

Figure 6: Pearson's correlation coefficients between features in the UCI air quality dataset.

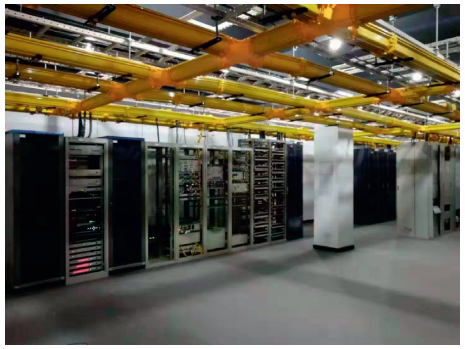

Figure 7: Base station where the data were collected.

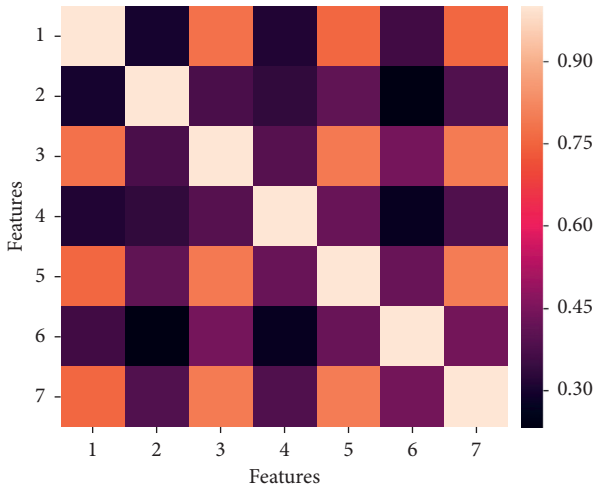

Figure 8: Pearson's correlation coefficients between features on BSS.

(5) The LSTM versions of RAEF generally outperformed the GRU versions.

(6) Focusing on the common-mode patterns, the RNN approaches clearly show how much greater the impact of this type of missing data is over the univariate brand. For example, looking at BSS with a high missing rate (20%–30%), the performance of all frameworks deteriorates rapidly.

In addition to these basic observations, we also noted some distinguishing performance features when comparing the gated regulator variants of RAEF to the plain version. In general, the gated regulator implementations of RAEF both outperformed the other frameworks within a limited range of missing rates and had obvious advantages under higher missing rates. From 5% to 15% missing rates on UAQ, the percentage increase in MRE over the non-gated versions of RAEF with the common-mode patterns was 7.46% and 23.54%, respectively. With the univariate pattern, this number was 8.57% and 20.23%. We can see the same trend with the BSS dataset. These results suggest that the gated regulator was able to reduce the negative impact of increased missing rates with both types of missing patterns.

5.2. Imputation Performance for Mixed Missing Pattern. Ideally, the missing data in a time series will conform to a single pattern—either univariate mode or common mode. However, there are occasions where both patterns will be present. For this series of experiments, we fixed the missing rates—at 10% for the UAQ dataset and at 20% for BSS. We then simulated the following patterns of missing data in the time series: 100% univariate, 20% common mode (CM)/80% univariate (UM), 40%CM/60%UM, 60%CM/40%UM, 80% CM/20%UM, and 100% CM. Figure 9 shows the results. RAEF-LSTM-GR was the clear performer with significantly better results than the others.

5.3. Task: Imputing Missing Values in an Incomplete Time Series. To more clearly show the importance of data imputation for downstream applications, we undertook a prediction task using incomplete time-series data and compared the results to the same task using imputed data. To approximate different real application scenarios, we performed the tasks with a range of missing rates. More specifically, we prepared versions of the UAQ dataset with missing rates of 5%, 10%, and 15% and conducted three groups of experiments A, B, and C as follows:

(1) A: impute with R AIN-LSTMF and then use an LSTM for prediction

(2) B: impute with BRITS and then use an LSTM for prediction

Table 2: Imputation performance on UAQ in MRE%.

MP	MR (%)	Baseline				Ours			
		BM (%)	KNN	LIME-LSTM (%)	BRITS (%)	RFL (%)	RFG (%)	RFLR (%)	RFGR (%)
CM	15	39.88	—	15.62	12.32	11.44	11.64	9.79	10.33
	10	29.24	—	14.16	10.58	10.02	10.51	9.32	9.71
	5	18.80	—	12.69	9.44	9.26	9.40	9.11	9.27
UM	15	45.28	32.10%	13.29	11.70	10.04	10.78	9.12	9.37
	10	32.47	23.84%	12.00	10.10	8.86	9.48	8.61	8.79
	5	19.35	15.88%	11.46	8.85	8.35	8.61	8.40	8.26

Table 3: Imputation performance on BSS in MRE%.

MP	MR (%)	Baseline				Ours			
		BM (%)	KNN	LIME-LSTM (%)	BRITS (%)	RFL (%)	RFG (%)	RFLR (%)	RFGR (%)
CM	30	60.62	—	24.20	17.84	16.99	17.06	14.06	14.32
	20	39.73	—	19.57	15.05	14.45	15.11	12.29	12.99
	10	16.32	—	15.43	14.14	12.98	13.58	11.28	11.67
UM	30	67.44	47.42%	19.11	15.91	14.12	14.01	11.97	12.34
	20	45.23	31.84%	14.28	13.26	12.28	12.26	10.94	11.52
	10	19.01	16.83%	11.85	11.73	10.88	11.47	9.94	10.01

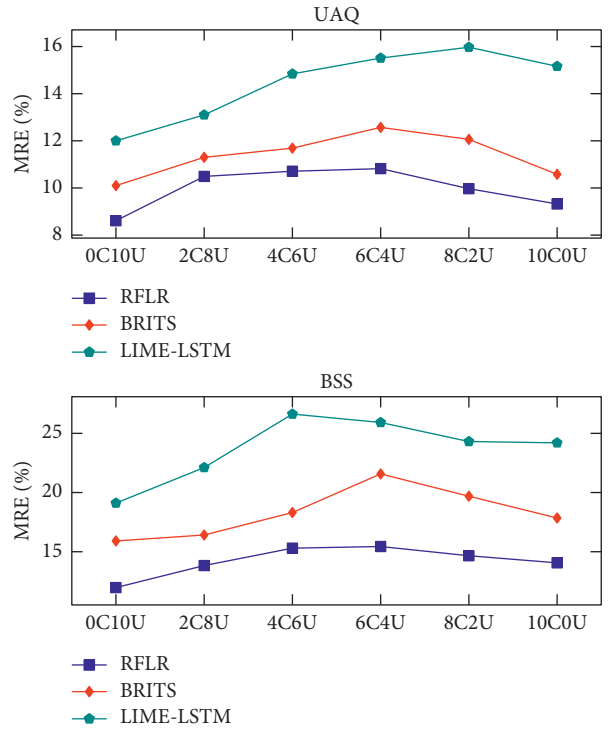

Figure 9: Imputation performance for the mixed missing pattern results in terms of MRE (%), where *a* C *b* U means that, at a fixed missing rate, *a* percent of the missing data followed a common-mode pattern and *b* percent followed a univariate pattern.

(3) C: impute with LIME-LSTM and then use an LSTM for prediction

Groups A-C all used the same LSTM for predictions, which contained 64 neurons and were trained with a complete dataset. The difference between the prediction results and the ground truth data was measured using MAE. Each experiment was repeated 50 times, and the results were recorded as shown in Figure 10.

5.4. The Choice of T. To assess choice process, we varied the value of T. As shown in Figure 11, RAEF-LSTM-GR generally delivered the best performance for each dataset. But, as the

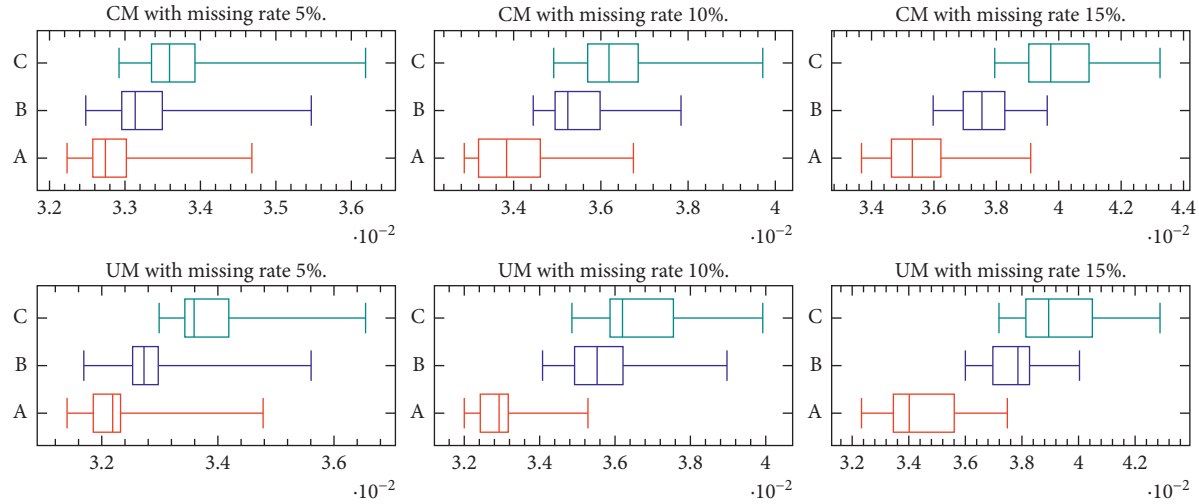

Figure 10: Plots show the prediction performance of three groups' results in terms of MAE.

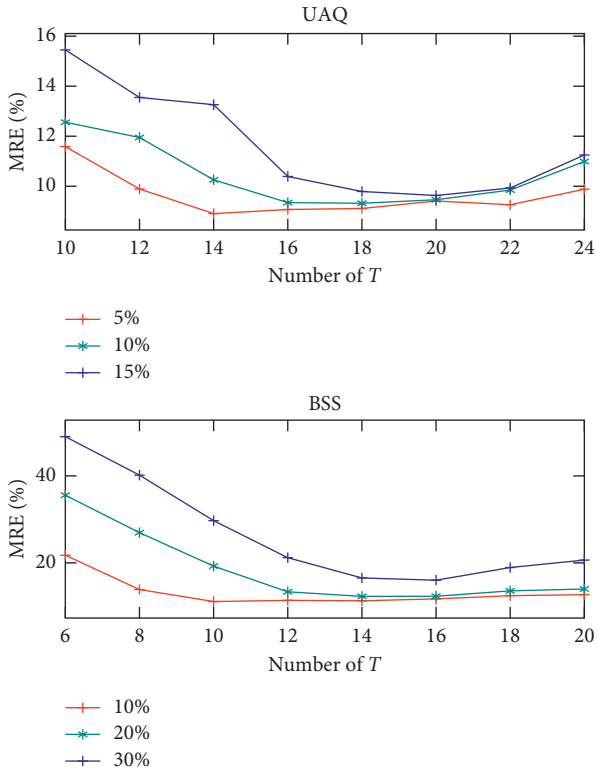

Figure 11: Imputation performance of the RAEF-LSTM-GR under different T results in MRE (%).

missing rate changed, the optimal value of T was slightly different. At higher rates, the RAEF-LSTM-GR preferred a larger T to obtain more information from the input time series. However, when T was too large, performance drops, indicating that the model was affected by an exploding gradient.

6. Conclusion

This study presents RAEF, an imputation framework for IIoT environments based on a recurrent autoencoder. RAEF identifies the missing patterns in incomplete time series and uses them as a guide to impute the values that are missing. As part of this research, we, for the first time, summarized the missing patterns in incomplete IIoT time-series data. Unlike some other methods, which decay the hidden state, RAEF uses a gated regulator to reduce the negative impact of larger missing rates. Tests on both synthetic and real data with this approach show that RAEF has greater robustness, more flexibility, and returns fewer errors than other state-of-the-art imputation frameworks designed for time-series data.

Data Availability

The BSS and UAQ data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Key R&D Program of China (2020YFB2010901) and in part by the Science and Technology Program of Zhejiang Province (no. 2020C01031).

References

[1] L. D. Xu, W. He, and S. Li, "Internet of things in industries: a survey," *IEEE Transactions on industrial informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.

[2] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: a survey," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 414–454, 2014.

[3] I. Mashal, T. Y. Chung, and O. Alsaryrah, "Toward service recommendation in internet of things," in *Proceedings of the The Seventh International Conference on Ubiquitous and*

Future Networks 2015 - ICUFN2015, Sapporo, Japan, July 2015.

[4] S. Raza, L. Wallgren, and T Voigt Svelte, "SVELTE: real-time intrusion detection in the internet of things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2661–2674, 2013.

[5] P. J. García-Laencina, J.-L. Sancho-Gómez, and A. R. Figueiras-Vidal, "Pattern classification with missing data: a review," *Neural Computing & Applications*, vol. 19, no. 2, pp. 263–282, 2010.

[6] M. L. Yadav and B. Roychoudhury, "Handling missing values: a study of popular imputation packages in r," *Knowledge-Based Systems*, vol. 160, pp. 104–118, 2018.

[7] C. Simolo, M. Brunetti, M. Maugeri, and T. Nanni, "Improving estimation of missing values in daily precipitation series by a probability density function-preserving approach," *International Journal of Climatology*, vol. 30, no. 10, pp. 1564–1576, 2010.

[8] L. Zhao, Z. Chen, Z. Yang, Y. Hu, and M. S. Obaidat, "Local similarity imputation based on fast clustering for incomplete data in cyber-physical systems," *IEEE systems journal*, vol. 12, no. 2, pp. 1610–1620, 2016.

[9] Y. Wei, Y. Tang, and P. D. McNicholas, "Flexible high-dimensional unsupervised learning with missing data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, vol. 42, no. 3, 2018.

[10] H. Demirtas, "Flexible imputation of missing data," *Journal of Statal Software*, vol. 85, 2018.

[11] B. S. Panda and R. Kumar Adhikari, "A method for classification of missing values using data mining techniques," in *Proceedings of the 2020 International Conference on Computer Science, Engineering and Applications (ICCSEA)*, pp. 1–5, Odisha, India, March 2020.

[12] M. R. Malarvizhi and A. S. Thanamani, "K-nearest neighbor in missing data imputation," *International Journal of Engineering Research and Development*, vol. 5, no. 1, 2013.

[13] Q. Zhang, L. T. Yang, Z. Chen, and F. Xia, "A high-order possibilistic C -means algorithm for clustering incomplete multimedia data," *IEEE Systems Journal*, vol. 11, no. 4, pp. 2160–2169, 2017.

[14] M. G. Rahman and M. Z. Islam, "Missing value imputation using a fuzzy clustering-based em approach," *Knowledge and Information Systems*, vol. 46, no. 2, pp. 389–422, 2016.

[15] P. S. Raja and K. Sasirekha, "A novel fuzzy rough clustering parameter-based missing value imputation," *Neural Computing & Applications*, vol. 32, no. 6, pp. 1–18, 2019.

[16] I. Arous, M. Khayati, P. Cudré-Mauroux, Y. Zhang, M. Kersten, and S Stalinlov, "Recovdb: accurate and efficient missing blocks recovery for large time series," in *Proceedings of the 2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pp. 1976–1979, IEEE, Macao, China, April 2019.

[17] I. P. S. Mary and L. Arockiam, "Imputing the missing data in iot based on the spatial and temporal correlation," in *Proceedings of the IEEE International Conference on Current Trends in Advanced Computing*, Bangalore, India, March 2017.

[18] S. Guastello, *Nonlinear Dynamical Systems Analysis for the Behavioral Sciences Using Real Data*, Routledge, Oxford, England, 2011.

[19] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial nets," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, ser. NIPS'14*, pp. 2672–2680, Montreal, Canada, December 2014.

[20] S. Yoon and S. Sull, "Generative adversarial multiple imputation network for highly missing data," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8453–8461, Seattle, WA, USA, June 2020.

[21] S. C. Li, B. Jiang, and B. M. Marlin, "Misgan: learning from incomplete data with generative adversarial networks," 2019, http://arxiv.org/abs/1902.09599.

[22] S. Yang, M. Dong, Y. Wang, and C. Xu, "Adversarial recurrent time series imputation," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–12, 2020.

[23] H. G. Kim, G. J. Jang, H. J. Choi, M. Kim, Y. W. Kim, and J. Choi, "Recurrent neural networks with missing information imputation for medical examination data prediction," in *Proceedings of the 2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 317–323, Jeju, South Korea, February 2017.

[24] L. E. E. Minseok, A. N. Jihoon, and L. E. E Younghee, "Missing-value imputation of continuous missing based on deep imputation network using correlations among multiple iot data streams in a smart space," *IEICE-Transactions on Info and Systems*, vol. 102, no. 2, 2019.

[25] Q. Ma, S. Li, L. Shen et al., "End-to-end incomplete time-series modeling from linear memory of latent variables," *IEEE transactions on cybernetics*, vol. 50, no. 12, 2019.

[26] L. Li, J. Zhang, Y. Wang, and B. Ran, "Missing value imputation for traffic-related time series data based on a multi-view learning method," *Intelligent Transportation Systems IEEE Transactions on*, vol. 20, no. 8, 2018.

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

[28] Z. C. Lipton, D. C. Kale, and R. C. Wetzel, "Directly modeling missing data in sequences with rnns: improved classification of clinical time series," 2016, http://arxiv.org/abs/1606.04130.

[29] W. Cao, D. Wang, J. Li, H. Zhou, Y. Li, and L. Li, "Brits: bidirectional recurrent imputation for time series," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems, ser. NIPS'18*, pp. 6776–6786, Curran Associates Inc., Red Hook, NY, USA, May 2018.

[30] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Scientific Reports*, vol. 8, no. 1, p. 6085, 2018.

[31] T. Feng and S. Narayanan, "Imputing missing data in large-scale multivariate biomedical wearable recordings using bidirectional recurrent neural networks with temporal activation regularization," in *Proceedings of the 2019 41st Annual International Conference of the IEEE Engineering in Medicine & Biology Society*, Berlin, Germany, July 2019.

[32] N. Anindita, H. A. Nugroho, and T. B. Adji, "A combination of multiple imputation and principal component analysis to handle missing value with arbitrary pattern," in *Proceedings of the 2017 7th International Annual Engineering Seminar (InAES)*, Yogyakarta, Indonesia, Auguest 2017.

[33] C.-F. Tsai and F.-Y. Chang, "Combining instance selection for better missing value imputation," *Journal of Systems and Software*, vol. 122, pp. 63–71, 2016.

[34] W. Insuwan, U. Suksawatchon, and J. Suksawatchon, "Improving missing values imputation in collaborative filtering with user-preference genre and singular value decomposition," in *Proceedings of the 2014 6th International Conference*

on Knowledge and Smart Technology (KST), pp. 87–92, Chonburi, Thailand, January 2014.

[35] S. Tak, S. Woo, and H. Yeo, "Data-driven imputation method for traffic data in sectional units of road links," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1762–1771, 2016.

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

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

[38] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, http://arxiv.org/abs/1406.1078.

[39] H. Wu, J. Hu, J. Sun, and D. Sun, "Edge computing in an iot base station system: reprogramming and real-time tasks," *Complexity*, vol. 2019, Article ID 4027638, 10 pages, 2019.

[40] T. Hastie, R. Tibshirani, and J. Friedman, "The elements of statistical learning," *Journal of the Royal Statal Society*, vol. 167, no. 1, p. 192, 2004.