**ABSTRACT** Time-series data with missing values are a common occurrence in various fields, including healthcare, meteorology, and robotics. The process of imputation aims to fill in the missing values with valid values. Most imputation methods implicitly train models due to the presence of missing values. In this paper, we propose Random Drop Imputation with Self-training (RDIS), a novel training method for time-series data imputation models. In RDIS, we generate extra missing values by applying a random drop to the observed values in incomplete data. We can explicitly train the imputation models by filling in the missing values. Moreover, we utilize self-training with pseudo values to exploit the original missing values. To enhance the quality of pseudo values, we set a threshold and filter them based on entropy calculation. To evaluate the effectiveness of RDIS for imputing time-series data, we test it across several imputation models and obtain competitive results on three real-world datasets.

**INDEX TERMS** Time series data imputation, self-training, semi-supervised learning, pseudo-value, ensemble learning.

**I. INTRODUCTION**

With technological advancements in sensors and other hardware, a large amount of time-series data can now be collected, providing valuable information for various environments such as IoT and autonomous driving [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13]. However, time-series data collection errors often occur due to sensor malfunction and human mistakes [14], leading to incomplete data with missing and observed values. One way to handle incomplete data is imputation, which involves filling in the missing values. Time series imputation aims to fill in the missing values in the incomplete time series data. However, most imputation works face the challenge of explicitly training the model due to the absence of ground truth for missing values. Recently, various imputation methods using deep learning [15], [16], [17], [18], [19], [20], [21], [22] have been developed, but they still rely on the reconstruction loss of observed values for model training. While this helps the learning stability of the imputation model, it is not directly learning about imputation. To address this limitation, we propose a novel training method called Random Drop Imputation with Self-training (RDIS). RDIS involves randomly dropping observed values in the incomplete time series data and training the model to impute these values.

Self-training [23], [24], [25], [26] is one of the methods in semi-supervised learning (SSL), which generates pseudo labels as ground truth of unlabeled data using pre-trained models trained with labeled data. Self-training can also be used for imputation by treating observed and missing values with labeled and unlabeled data. It is essential to use high-quality pseudo labels for proper self-training. In general, we can evaluate the reliability of pseudo labels with deterministic probabilities in the image classification [27], [28] and the semantic segmentation [29]. However, imputation is not a deterministic task; it is not easy to measure the reliability of the pseudo label generated from the pre-trained model. In this light, we generate several imputed values for one missing value, which helps the learning stability of the imputation model, and it is not directly learning about imputation. To address this limitation, we propose a novel training method called Random Drop Imputation with Self-training (RDIS). RDIS involves randomly dropping observed values in the incomplete time series data and training the model to impute these values.

Self-training [23], [24], [25], [26] is one of the methods in semi-supervised learning (SSL), which generates pseudo labels as ground truth of unlabeled data using pre-trained models trained with labeled data. Self-training can also be used for imputation by treating observed and missing values with labeled and unlabeled data. It is essential to use high-quality pseudo labels for proper self-training. In general, we can evaluate the reliability of pseudo labels with deterministic probabilities in the image classification [27], [28] and the semantic segmentation [29]. However, imputation is not a deterministic task; it is not easy to measure the reliability of the pseudo label generated from the pre-trained model. In this light, we generate several imputed values for one missing value, which helps the learning stability of the imputation model, and it is not directly learning about imputation. To address this limitation, we propose a novel training method called Random Drop Imputation with Self-training (RDIS). RDIS involves randomly dropping observed values in the incomplete time series data and training the model to impute these values.

The associate editor coordinating the review of this manuscript and approving it for publication was Prakash Periasamy. 

100720
value using multiple models and evaluate the confidence of the pseudo label with their entropy.

RDIS is composed of two parts: random drop imputation (RDI) and self-training. RDI randomly removes the observed values in the incomplete data and trains an imputation model to impute these values. Thus, unlike the previous methods, which reconstruct observed values in original data in the training process, RDI explicitly trains an imputation model. Furthermore, numerous augmented data can be generated by the random drop. To utilize these augmented data, RDI can employ ensemble learning.

Although incomplete data consists of missing and observed values, RDI only utilizes the observed values to train the imputation model because missing values do not have ground truth. To deal with this issue, RDIS is designed, which adds self-training to RDI to train the imputation model utilizing the missing values in the original incomplete data. Self-training is part of SSL using pseudo labels generated from a teacher (pre-trained) model on unlabeled data. Self-training with the pseudo label is usually used in deterministic tasks, but we propose pseudo values instead of pseudo labels since we aim to generate continuous values. RDIS has three main steps: 1) train the teacher (pre-trained) models using RDI, 2) use the teachers to generate pseudo values on missing values, and 3) train each model on the combination of observed values and pseudo values. Then, not every pseudo value is reliable, so we calculate the entropy of the pseudo values to measure the confidence of each pseudo value. To the best of our knowledge, this paper represents the first attempt to utilize self-training in the task of imputation.

In summary, the contributions of this paper are three-fold:

- We propose RDI for explicit training of imputation models. We utilize random drop data, where observed values are randomly omitted from incomplete data, providing ground truth for missing values. RDI distinctly employs this random drop data to explicitly train imputation, setting it apart from prior training approaches.
- We propose RDIS, the pioneering attempt to incorporate self-training into time-series imputation. RDIS synergizes self-training and RDI, harnessing missing values to generate pseudo values. We further introduce a process to evaluate the reliability of pseudo values, tailored for stable self-training in time-series imputation.
- We evaluate our method on three multivariate real-world datasets: air quality, gas sensor, and electricity. Experimental results demonstrate that imputation accuracy improves when our method is applied to various imputation models.

II. RELATED WORKS

Since imputation methods using machine learning [30], [31], [32], [33] have surged and have shown significant advantages, deep learning implementation has been studied in imputation tasks, which have recently achieved more developments. In particular, in the case of time-series data, imputation methods using the RNNs [18], [19], [21] have gained much attention. Reference [18] developed GRU-D, adding a decay term to the GRU cell so that the model can learn the effect of the temporal gap between missing data. Reference [21] used bidirectional RNN to improve imputation performance. Inspired by the residual network (ResNet) [19], [34] added a weighted linear memory vector to RNN to make the model less affected by missing data. Indirect learning was used in deep-learning-based studies by training a model only with observed values from incomplete data [18], [21] or training with classification loss [19]. In addition, transformer-based models have also been studied recently. SAITS [10] proposed a diagonally-masked self-attention block to generate missing values effectively. MTSIT [6] proposed a way to train a transformer architecture in an unsupervised manner.

Imputation methods using generative models [13], [16], [17], [22], [35] also have made significant advances. GAN [36] guides a model to learn the distribution of the original data and generate data with a similar distribution, thereby allowing the model to learn imputation. Reference [16] adopted conditional GAN by using the original data and mask as the condition. Reference [22] designed a recurrent network using GRU-I, a simplified version of GRU-D, and applied GAN. Similarly, [17] used GRU-I but designed a recurrent autoencoder, which is the main difference. Unlike the existing autoregressive methods, [35] developed a method training in a non-autoregressive fashion. Reference [35] also uses the divide and conquer strategy to fill in the missing values gradually rather than filling in at once. Reference [13] adopted a conditional score-based diffusion model [37] for imputation. The conditional diffusion model is explicitly trained for imputation and exploits correlations between observed values.

Semi-supervised learning (SSL) is a class of algorithms to train with labeled and unlabeled data [23], [24], [38], [39]. One of the popular approaches for SSL is self-training, which uses an ensemble or a teacher model to generate pseudo labels for the unlabeled data. Recently, self-training has been introduced in various fields, such as image classification [27], [28], semantic segmentation [29], unsupervised domain adaptation [40], [41]. In the imputation task, we can treat missing values as unlabeled data and observed values as labeled data since missing values do not have ground truth. From this perspective, we adopt self-training for the imputation task.

III. METHOD

RDIS consists of RDI and self-training, where the former trains an imputation model explicitly using observed values of incomplete data, while the latter uses reliable pseudo values for more accurate training. Self-training employs an imputation model trained with RDI as a teacher model. In the following, we delineate the problem statement, RDI, and self-training.

A. PROBLEM STATEMENT

We denote an original incomplete time-series data as

$$X = \{x_1, \ldots, x_T\} \in \mathbb{R}^{T \times D},$$

where each element is a vector

$$x_t = \{x_t^1, \ldots, x_t^D\} \in \mathbb{R}^D.$$  

Since X is an incomplete
time-series data, we define a mask, \( M \in \{0, 1\}^{T \times D} \), to indicate the location of the missing values. Each element of the \( M \) is expressed as follows:

\[
m_i^d = \begin{cases} 
0, & \text{if } x_i^d \text{ is a missing value} \\
1, & \text{otherwise}
\end{cases}
\] (1)

We also introduce a complete time-series data, \( X_{\text{ideal}} \in \mathbb{R}^{T \times D} \), which has hypothetical values considered as ground truth. Then, we can denote \( X = X_{\text{ideal}} \odot M \), where \( \odot \) indicates element-wise multiplication. By using these notations, imputation attempts to find \( \theta \) that minimizes the following objective function:

\[
\min_{\theta} \mathbb{E}[||X_{\text{ideal}} \odot (1 - M) - F(X; \theta) \odot (1 - M)||_2],
\] (2)

where \( F \) is an imputation model with parameter \( \theta \). Note that optimizing (2) is a challenging task because \( X_{\text{ideal}} \) is not existing. Various approaches were taken to solve this problem in the previous imputation studies, and the most common approach is to use the following reconstruction loss:

\[
L_{\text{recon}} = ||X \odot M - F(X; \theta) \odot M||_2,
\] (3)

which is a loss term for reconstructing observed values. (3) can help imputation, but it is different from (2), which learns to fill missing values directly. Therefore, training imputation model using (3) can be called implicit training. In contrast, we propose a new training method using a random drop imputation (RDI) for explicit training.

**B. RANDOM DROP IMPUTATION**

RDI explicitly trains an imputation model by random drop data obtained from randomly removing observed values in the time-series data. Let’s denote the random drop data as \( \tilde{X} = \{\tilde{x}_1, \ldots, \tilde{x}_N\} \in \mathbb{R}^{N \times D} \), where each element is a vector \( \tilde{x}_i = \{\tilde{x}_{i1}, \ldots, \tilde{x}_{iD}\} \in \mathbb{R}^D \). We also denote a mask which represents the location of both originally missing values and randomly dropped values, as \( \tilde{M} \) of which element \( \tilde{m}_i^d \) is defined as follows:

\[
\tilde{m}_i^d = \begin{cases} 
0, & \text{if } \tilde{x}_i^d \text{ is a missing value} \\
1, & \text{otherwise}
\end{cases}
\] (4)

The loss function of RDI, called imputation loss, can be expressed as follows:

\[
L_{\text{impute}}(\tilde{X}, \tilde{M}) = ||X \odot (M - \tilde{M}) - F(\tilde{X}; \theta) \odot (M - \tilde{M})||_2 + ||\tilde{X} \odot \tilde{M} - F(\tilde{X}; \theta) \odot \tilde{M}||_2,
\] (5)

where the first term of (5) is the core objective function of imputation, which is similar to (2). After the random drop, missing values with ground truth are generated, which can be explicit learning of (2). In contrast, the second term is the reconstruction loss of observed values, which helps stable training.

Note that as our proposed method, RDI, creates missing values with ground truth, using the first term of (5), we can explicitly train an imputation model. In addition, RDI has one more advantage in terms of data augmentation, in which we can generate various combinations to drop observed values randomly from the original data. By taking this advantage, we generate multiple random drop data from one original data and use them for RDI. Also, to utilize the augmented data, our RDI employs ensemble learning.

To deal with unstable and over-fitting issues, bootstrap with ensemble learning [42], [43] was applied to the imputation task. This motivates us to adopt ensemble learning to boost imputation performance. Bootstrap constructs several sets using random sampling and trains models with each set. However, one of the disadvantages of bootstrap is the bias from the random sampling. On the other hand, as RDI creates different sets by augmenting the entire dataset, we can apply ensemble learning without bias.

As shown in Figure 1(a), we generate \( N \) different random drop data, denoted as \( \tilde{X} = \{\tilde{x}_1, \ldots, \tilde{x}_N\} \). Then, \( N \) imputation
models, which are composed of the same structure but trained with different data using the loss function (5) can be denoted as $E = \{F_1, \ldots, F_N\}$. Each model $F_k$ is trained to generate original data $X$ from random drop data $\hat{X}_k$. After training, the original data is fed to each pre-trained model $F_k$ for imputation. However, the output values from individual models are all distinctive since differently augmented data are used when training individual models. Therefore, the average of $N$ output values is considered a final output of the ensemble model.

**C. RANDOM DROP IMPUTATION WITH SELF-TRAINING**

Original incomplete data $X$ contains observed and missing values (without ground truth). RDI only exploits observed values of the original incomplete data. Hence, we apply self-training to exploit the missing values as well as the observed values. We utilize pseudo values of originally missing values for self-training. We feed the original data to the pre-trained models, $F_1, \ldots, F_N$, for generating the pseudo values. Then, the average outputs of the pre-trained models, $\hat{X} = \frac{1}{N} \sum_{k=1}^{N} F_k(X)$ are considered as pseudo values. The pseudo values are updated periodically every fixed number of epochs, which we call an update epoch. The loss function of each model $F_k$ for the self-training is expressed as follows:

$$L_{self,k}(\hat{\mathbf{X}}, \hat{\mathbf{X}}_k) = || \hat{\mathbf{X}} \odot (1 - \mathbf{M}) - F_k(\hat{\mathbf{X}}_k; \theta_k) \odot (1 - \mathbf{M}) ||_2 + || \mathbf{X} \odot \mathbf{M} - F_k(\hat{\mathbf{X}}_k; \theta_k) \odot \mathbf{M} ||_2,$$

where the first term represents the loss between missing and pseudo values, and the second term denotes the reconstruction loss of the observed values.

RDIS also works better with an additional filtering rule. We filter out pseudo values that have low confidence. The evaluation of the confidence of pseudo values is different from that of pseudo labels. Usually, most self-training methods filter the pseudo labels using predicted probability from the pre-trained model, but there is no probability in the imputation. Considering this issue, RDIS uses entropy to check whether each pseudo value is reliable or not.

In the case of classification and segmentation, which predict discrete labels, the model’s output is a probability of $[0,1]$. Therefore, a deterministic task can easily measure the confidence of a pseudo-label. In contrast, the imputation model predicts continuous values, so the outputs are not a probability. To this end, entropy is employed to calculate the confidence level. We apply ensemble learning to RDI, where $N$ pre-trained models generate $N$ values for each missing value in the original incomplete data. Assuming that these $N$ values follow the normal distribution, the entropy can be calculated as follows:

$$\text{Entropy} = -\int_{-\infty}^{\infty} p(x) \log p(x) dx = \ln(\sigma \sqrt{2\pi e}),$$

The variance of the $N$ values, $\sigma$, is employed to measure the entropy. The output value with low entropy is considered reliable and used as a pseudo value, whereas the output value with high entropy is considered uncertain and discarded. The threshold value that distinguishes reliable/unreliable entropy is determined empirically through experiments. The overall procedure of selecting pseudo values is shown in Figure 2. For calculation convenience, we set the pseudo value threshold as the variance of the $N$ values. The pseudo-code of the proposed RDIS is illustrated in Algorithm 1.

**Algorithm 1 Training Strategy of RDIS**

**Input:** Incomplete time series data $X$, mask $M$, pre-trained imputation models $F_1(X; \theta_1), \ldots, F_N(X; \theta_N)$ from RDI

**Parameters:** $N$, $\tau$, numEpochs, numUpdateEpochs

**Output:** Trained $N$ imputation models $F_1, \ldots, F_N$

1. $e \leftarrow 0$
2. **while** $e < \text{numEpochs}$ **do**
3. **if** $\% \text{numUpdateEpochs} == 0$ **then**
4. Empty list containing pseudo values: $L \leftarrow []$
5. **for** $i=1, \ldots, N$ **do**
6. $L \leftarrow \text{Concatenate}(L; F_i(X, \theta_i))$ // final dimension of $L : (N, T, D)$
7. **end**
8. **for** $t=1, \ldots, T$ **do**
9. **for** $d=1, \ldots, D$ **do**
10. Compute variance $\sigma$ of time step $t$ and dimension $d$ using data in $L$
11. **if** $\sigma < \tau$ **then**
12. $X^d_t \leftarrow \frac{1}{N} \sum_{i=1}^{N} L_{i,t,d}$ // generate pseudo values
13. **end**
14. **end**
15. **end**
16. **end**
17. **for** $i=1, \ldots, N$ **do**
18. $F_i \leftarrow L_{self,i}(\hat{\mathbf{X}}, \hat{\mathbf{X}}_i)$ // update each model using pseudo values
19. **end**
20. $e \leftarrow e + 1$
21. **end**
TABLE 1. The performance comparison of imputation methods under different missing rates on the air quality dataset. Metrics are reported in the order of MSE / MAE.

| Method | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% |
|--------|-----|-----|-----|-----|-----|-----|-----|-----|
| Forward | 0.247 / 0.186 | 0.264 / 0.193 | 0.271 / 0.204 | 0.269 / 0.216 | 0.275 / 0.237 | 0.326 / 0.260 | 0.389 / 0.300 | 0.443 / 0.350 |
| Backward | 0.185 / 0.180 | 0.214 / 0.191 | 0.285 / 0.207 | 0.283 / 0.221 | 0.280 / 0.238 | 0.333 / 0.262 | 0.362 / 0.295 | 0.445 / 0.349 |
| MICE [32] | 0.341 / 0.233 | 0.414 / 0.272 | 0.389 / 0.293 | 0.424 / 0.322 | 0.431 / 0.354 | 0.566 / 0.379 | - | - |
| KNN [44] | 0.152 / 0.174 | 0.232 / 0.196 | 0.285 / 0.239 | 0.350 / 0.286 | 0.393 / 0.324 | 0.433 / 0.338 | 0.484 / 0.370 | 0.578 / 0.437 |
| GRU [45] | 0.314 / 0.322 | 0.362 / 0.337 | 0.376 / 0.351 | 0.374 / 0.370 | 0.383 / 0.384 | 0.433 / 0.403 | 0.452 / 0.433 | 0.496 / 0.463 |
| Bi-GRU [46] | 0.271 / 0.299 | 0.314 / 0.312 | 0.334 / 0.327 | 0.329 / 0.336 | 0.336 / 0.356 | 0.390 / 0.379 | 0.403 / 0.398 | 0.445 / 0.432 |
| M-RNN [20] | 0.158 / 0.181 | 0.213 / 0.204 | 0.234 / 0.221 | 0.232 / 0.236 | 0.242 / 0.257 | 0.293 / 0.283 | 0.325 / 0.332 | 0.404 / 0.395 |
| BRITS [21] | 0.166 / 0.158 | 0.208 / 0.168 | 0.221 / 0.184 | 0.209 / 0.198 | 0.214 / 0.218 | 0.266 / 0.244 | 0.289 / 0.287 | 0.342 / 0.334 |
| MTSIT [6] | 0.337 / 0.344 | 0.401 / 0.361 | 0.417 / 0.374 | 0.412 / 0.387 | 0.432 / 0.403 | 0.467 / 0.418 | 0.490 / 0.448 | 0.556 / 0.488 |
| SAITS [10] | 0.202 / 0.215 | 0.260 / 0.234 | 0.296 / 0.262 | 0.306 / 0.285 | 0.314 / 0.300 | 0.377 / 0.335 | 0.394 / 0.354 | 0.426 / 0.380 |
| RDIS (GRU) | 0.185 / 0.239 | 0.238 / 0.252 | 0.260 / 0.267 | 0.258 / 0.286 | 0.263 / 0.301 | 0.308 / 0.320 | 0.328 / 0.344 | 0.383 / 0.380 |
| RDIS (Bi-GRU) | 0.141 / 0.212 | 0.181 / 0.227 | 0.201 / 0.234 | 0.198 / 0.250 | 0.204 / 0.264 | 0.253 / 0.285 | 0.267 / 0.308 | 0.318 / 0.338 |
| RDIS (M-RNN) | 0.143 / 0.174 | 0.170 / 0.192 | 0.199 / 0.205 | 0.203 / 0.221 | 0.217 / 0.241 | 0.272 / 0.268 | 0.302 / 0.303 | 0.354 / 0.366 |
| RDIS (BRITS) | 0.130 / 0.153 | 0.168 / 0.163 | 0.191 / 0.173 | 0.184 / 0.185 | 0.187 / 0.212 | 0.244 / 0.236 | 0.270 / 0.277 | 0.335 / 0.325 |

We employed the previous machine-learning methods and deep-learning-based methods. We evaluated the performance on three real-world time series datasets: air quality, gas sensor, and electricity datasets.

A. DATASET DESCRIPTION

In our experiments, we utilized three real-world time series datasets. The following provides a detailed description of the datasets.

1) AIR QUALITY DATA

Beijing Multi-Site Air-Quality dataset [47] from the UCI machine learning repository, abbreviated as the air quality dataset, was used in the experiment. The air quality dataset is air pollutant data from 12 nationally-controlled air-quality monitoring sites. This measurement was collected hourly from 03/01/2013 to 02/28/2017. The dataset consists of 12 features. Among the features, we excluded the direction of the wind because it is implausible to represent the value numerically. We randomly selected 48 consecutive time steps to generate one time series. The original missing rate of the air quality dataset is 1.3%.

2) GAS SENSOR DATA

Gas Sensor Array Temperature Modulation dataset [48] from the UCI machine learning repository, abbreviated as gas sensor dataset, was also used in the experiment. The gas sensor dataset comprises 19 features, including humidity, CO (ppm), and temperature. We randomly select 48 consecutive time steps to generate one time series. The gas sensor dataset has no missing values.

3) ELECTRICITY DATA

Electricity dataset [49] from the UCI machine learning repository was used for evaluating performance in the large feature dimension dataset. The electricity dataset is electricity consumption data collected from 370 clients. This measurement was collected every 15 minutes from 01/01/2011 to 12/31/2014. This dataset contains 370 features.

We randomly selected 100 consecutive time steps to generate one time series. The electricity dataset has no missing values.

B. BASELINE METHODS

For the purpose of imputation performance comparison, we used the following baselines:

- **Forward/Backward**: The missing values are simply imputed with the last forward/backward observed values.
- **MICE**: The incomplete data are estimated with a low-rank approximation, and the missing values are filled with the singular vectors [32].
- **KNN**: k-nearest neighbors (KNN) [44] imputes the missing values with the weighted average of the \( k \) nearest sequences.
- **GRU/Bi-GRU**: The uni/bidirectional autoregressive Gated Recurrent Unit (GRU) [45], [46] model is for imputing missing values. The hidden states are fed to a fully connected layer to obtain the output.
- **M-RNN**: M-RNN [20] uses a bidirectional RNN. M-RNN estimates the missing values using the hidden states across streams in addition to within streams.
- **BRITS**: BRITS [21] imputes missing values using a bidirectional RNN and considers correlation among different missing values.
- **MTSIT**: MTSIT [6] is a transformer-based imputation model in an unsupervised scheme for multivariate time-series imputation.
- **SAITS**: SAITS [10] is based on a multi-head self-attention module and imputes the missing values using weighted combination blocks.

We used Impyute\(^1\) to implement MICE and fancyimpute\(^2\) to implement KNN.

C. EXPERIMENTAL SETTINGS

For efficient and stable training, we applied Z-score normalization to standardize all datasets with zero mean and

\(^1\)https://github.com/eltonlaw/impyute
\(^2\)https://github.com/iskandr/fancyimpute
TABLE 2. The performance comparison of imputation methods under different missing rates on the gas sensor dataset. Metrics are reported in the order of MSE / MAE.

| Method  | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% |
|---------|-----|-----|-----|-----|-----|-----|-----|-----|
| Forward | 0.083 / 0.098 | 0.099 / 0.107 | 0.118 / 0.118 | 0.146 / 0.133 | 0.185 / 0.155 | 0.240 / 0.185 | 0.322 / 0.232 | 0.474 / 0.315 |
| Backward| 0.084 / 0.098 | 0.102 / 0.108 | 0.127 / 0.118 | 0.152 / 0.133 | 0.189 / 0.155 | 0.245 / 0.185 | 0.331 / 0.232 | 0.550 / 0.317 |
| MICE [32] | 0.074 / 0.098 | 0.082 / 0.107 | 0.109 / 0.131 | 0.147 / 0.187 | 0.215 / 0.253 | - | - | - |
| KNN [44] | 0.035 / 0.031 | 0.036 / 0.055 | 0.039 / 0.062 | 0.127 / 0.088 | 0.242 / 0.160 | 0.377 / 0.259 | 0.492 / 0.330 | 0.801 / 0.426 |
| GRU [45] | 0.073 / 0.168 | 0.084 / 0.182 | 0.094 / 0.201 | 0.113 / 0.218 | 0.135 / 0.242 | 0.165 / 0.274 | 0.200 / 0.317 | 0.322 / 0.375 |
| Bi-GRU [46] | 0.066 / 0.160 | 0.077 / 0.189 | 0.086 / 0.197 | 0.102 / 0.215 | 0.124 / 0.236 | 0.145 / 0.259 | 0.180 / 0.297 | 0.290 / 0.352 |
| M-RNN [20] | 0.033 / 0.080 | 0.040 / 0.098 | 0.038 / 0.093 | 0.039 / 0.105 | 0.041 / 0.106 | 0.043 / 0.109 | 0.043 / 0.111 | 0.102 / 0.136 |
| BRITS [21] | 0.021 / 0.059 | 0.023 / 0.065 | 0.023 / 0.066 | 0.028 / 0.078 | 0.034 / 0.083 | 0.041 / 0.101 | 0.052 / 0.112 | 0.160 / 0.142 |
| MTSIT [6] | 0.086 / 0.184 | 0.090 / 0.194 | 0.128 / 0.232 | 0.127 / 0.229 | 0.155 / 0.249 | 0.164 / 0.258 | 0.218 / 0.299 | 0.357 / 0.360 |
| SAITS [10] | 0.028 / 0.080 | 0.030 / 0.087 | 0.027 / 0.081 | 0.028 / 0.079 | 0.031 / 0.085 | 0.033 / 0.087 | 0.032 / 0.086 | 0.087 / 0.105 |

D. PERFORMANCE COMPARISONS STUDIES

In the case of the air quality and gas sensor datasets, we allocated 50% of the data for training, 25% for validation, and the remaining 25% for testing in sequential order. For the electricity dataset, the first 10 months of data were used for testing, followed by the subsequent 10 months for validation, while the remaining data was utilized for training.

For all datasets, we utilized the following common model training parameters in our experiments: an input batch size of 128, the Adam optimizer, a learning rate of $5 \times 10^{-4}$, and a total of 2000 training epochs. The dimension of the hidden unit was set to 100 for the air quality and gas sensor datasets, while for the electricity dataset, it was set to 200 for all RNN-based methods.

We empirically tuned the hyper-parameters: the random drop rate, the threshold of the pseudo value, and the pseudo value update epoch. Each experiment was repeated five times, and the imputation performance was evaluated using both of the mean square error (MSE) and mean absolute error (MAE) metrics. We evaluated the imputation performance across different missing rates to demonstrate the robustness of RDIS. As the original missing values cannot measure MSE due to the lack of ground truth, we randomly discarded a percentage ($p\%$) of the observed data and used the discarded portion as the ground truth. The value of $p$ was selected from the set {10, 20, 30, 40, 50, 60, 70, 80}. In the testing stage of RDIS, we obtained the final output by averaging the predictions of the $N$ models used in ensemble learning.
Due to the limitation of searching hyper-parameters for every combination of missing rates, models, and datasets, we conducted an empirical hyper-parameter search based on each dataset. These hyper-parameters were determined through the empirical search for each dataset, as described in Section IV-G. Throughout the training cycles, all parameters remained fixed.

Since BRITS and M-RNN were specifically designed for time series imputation, they consistently achieved lower imputation errors than GRU-based models across all missing rates. However, when RDIS was applied to the GRU-based models, they exhibited competitive performance comparable to BRITS and M-RNN. Furthermore, when RDIS was applied to all models, we consistently observed lower errors across all missing rates. This finding highlights the versatility and effectiveness of RDIS in training imputations across different models. Particularly noteworthy is the combination of RDIS with BRITS, which consistently achieved the best performance in terms of MAE and outperformed most cases in MSE. Additionally, in the electricity dataset, the application of RDIS resulted in performance improvement across all models. This demonstrates the efficient training capability of RDIS, even on datasets with large feature dimensions, in contrast to the air quality and gas sensor datasets.

E. ABLATION STUDIES

To analyze the contribution of RDIS, we grafted it onto GRU and Bi-GRU, and we conducted experiments on four different settings: the model without RDIS (None), the model with RDI but without ensemble learning (RDI/E), the model with RDI, and the model with RDIS. We used the air quality and gas sensor dataset under different missing rates, and the same hyper-parameters in Table 2. Figure 3 exhibits the effects of RDI/E, RDI, and RDIS. First, all our methods dramatically improved in all cases. It implies that learning with our methods led to properly training the imputation models. Also, RDI showed better performance than RDI/E. It demonstrates the capability of ensemble learning. Our final observation is that RDIS generally showed the best performance, and the higher the missing rate, the higher the performance improvement. As the missing rate increases, an imputation model quickly overfits due to a lack of data. Nevertheless, RDIS can prevent overfitting by replacing missing values with reliable pseudo values.

F. PERFORMANCE COMPARISONS ON THE SPECIFIC MISSING PATTERN

To show the performance of RDIS under the different missing patterns, we evaluated several models in the specific condition. The random missing pattern is suitable for evaluating imputation performance but is rare in real practices. Therefore, we generated the periodic missing pattern. When \( o \) is expressed as observed and \( m \) as the missing value, we created a cyclical missing pattern in which \( o, o, m, m \) repeats in the gas sensor dataset. We used GRU, Bi-GRU, M-RNN, and BRITS for the experiment and the same hyper-parameters in Table 2. As shown in Table 4, all imputation models with RDIS also outperformed under the periodic missing pattern. This demonstrates that RDIS can be used with various missing patterns as well as random missing patterns.

G. HYPER-PARAMETER STUDIES

In this section, we empirically analyzed hyper-parameters (random drop rate, threshold of pseudo value, and pseudo value update epoch), which control the imputation performance. We used hyper-parameters searched in this section for experiments in Section IV-D.

1) RANDOM DROP RATE ANALYSIS

We analyzed the effect of the random drop rate (RDR) using two datasets. For this analysis, we used Bi-GRU with RDI, and each dataset was tested for a missing rate of 50%. Table 5 summarizes experimental results on the imputation performances for RDRs of 10%, 20%, 30%, 40%, and 50%.

---

**TABLE 4. Performance comparisons under the periodic missing pattern.**

| Model      | MSE   | MAE   | Model      | MSE   | MAE   |
|------------|-------|-------|------------|-------|-------|
| GRU        | 0.441 | 0.431 | M-RNN      | 0.243 | 0.242 |
| RDIS (GRU) | 0.311 | 0.337 | RDIS (M-RNN)| 0.131 | 0.165 |
| Bi-GRU     | 0.355 | 0.378 | BRITS      | 0.134 | 0.088 |
| RDIS (Bi-GRU) | 0.176 | 0.217 | RDIS (BRITS)| 0.108 | 0.060 |

**TABLE 5. Analysis of random drop rate on RDI.**

| Dataset     | 10%  | 20%  | 30%  | 40%  | 50%  |
|-------------|------|------|------|------|------|
| Air Quality | 0.250| 0.212| 0.209| 0.219| 0.262|
| Gas sensor  | 0.035| 0.028| 0.035| 0.047| 0.071|
In the air quality dataset, the best performance was obtained when RDR was 30%, and in the gas sensor dataset, the best performance was obtained at 20%.

2) PSEUDO VALUE ANALYSIS
In order to select reliable pseudo values, we measured the entropy of each pseudo value. To verify our proposed reliability measurement method, we computed the pseudo value’s accuracy according to entropy. The entropy is proportional to \( \ln(\sigma) \), as shown in (7), so we report the pseudo value’s accuracy according to the variance for convenience of calculation. We used the air quality dataset under different missing rates for this experiment. We also used Bi-GRU with RDIS and fixed an update epoch as 400. Figure 4 (a) shows that the pseudo value’s accuracy declines when the variance increases. It means that pseudo values with low variance are reliable ones, which verifies the effectiveness of our proposed pseudo value selection method.

3) UPDATE EPOCH ANALYSIS
We updated pseudo values periodically at a fixed epoch. To evaluate the effect of the pseudo value update epoch on RDIS, we experimented with update epochs of 100, 400, 1000, and 2000, respectively. In this experiment, Bi-GRU with RDIS was employed on the air quality dataset under different missing rates, and a variance of 0.03 was used as a threshold for selecting the pseudo value. We report the rate of reduction in MSE of RDIS compared to that of RDI. Figure 4 (b) shows the result of this experiment. Frequent updates of pseudo values have a detrimental impact on imputation performance. This stems from the frequent shifts in the convergence direction of the model. Conversely, delaying the update of pseudo values can also lead to adverse consequences. This is because the model, through self-training, generates more precise pseudo values as it improves its performance. Therefore, we tried to find the appropriate update epoch of pseudo values. As indicated by Figure 4 (b), a performance enhancement is observed across all missing rates when the update epoch is set to 400. Consequently, we chose to utilize an update epoch of 400 in our experiment.

V. CONCLUSION
This paper proposed a novel explicit training method, RDIS, which combines RDI and self-training to impute incomplete time-series data. RDIS learns imputation explicitly via random drop data augmented from the original data. To fully leverage the random drop data, RDI employs ensemble learning, which significantly enhances imputation performance. Moreover, self-training is incorporated into RDI to design RDIS, which is first introduced in imputation. We evaluated the performance of the three real-world datasets. Experimental results show that the bidirectional GRU with RDIS achieved state-of-the-art results with a significant margin. For future work, we will explore the application of RDIS to tabular data imputation. Additionally, we plan to investigate the inclusion of a loss function for pseudo values with high uncertainty levels.

ACKNOWLEDGMENT
(Tae-Min Choi and Ji-Su Kang are co-first authors.)

REFERENCES
[1] B. Batres-Estrada, “Deep learning for multivariate financial time series,” M.S. thesis, KTH Math. Statist., 2015.
[2] S. Bauer, B. Scholkopf, and J. Peters, “The arrow of time in multivariate time series,” in Proc. Int. Conf. Mach. Learn., 2016, pp. 2043–2051.
[3] F. Iglesias and W. Kastner, “Analysis of similarity measures in times series clustering for the discovery of building energy patterns,” Energies, vol. 6, no. 2, pp. 579–597, Jan. 2013.
[4] K. Hayirinen, K. Saranto, and P. Nykänen, “Definition, structure, content, use and impacts of electronic health records: A review of the research literature,” Int. J. Med. Inform., vol. 77, no. 5, pp. 291–304, May 2008.
[5] X. Tang, H. Yao, Y. Sun, C. C. Aggarwal, P. Mitra, and S. Wang, “Joint modeling of local and global temporal dynamics for multivariate time series forecasting with missing values,” in Proc. AAAI, 2020, pp. 5956–5963.
[6] A. Y. Yildiz, E. Koç, and A. Koç, “Multivariate time series imputation with transformers,” IEEE Signal Process. Lett., vol. 29, pp. 2517–2521, 2022.
[7] Y. Qian, L. Tian, B. Zhai, S. Zhang, and R. Wu, “Infomer-WGAN: High missing rate time series imputation based on adversarial training and a self-attention mechanism,” Algorithms, vol. 15, no. 7, p. 252, Jul. 2022.
[8] R. Wang, Z. Zhang, Q. Wang, and J. Sun, “TLGRU: Time and location gated recurrent unit for multivariate time series imputation,” EURASIP J. Adv. Signal Process., vol. 2022, no. 1, p. 74, Sep. 2022.
[9] J. Zhang, X. Mu, J. Fang, and Y. Yang, “Time series imputation via integration of revealed information based on the residual shortcut connection,” IEEE Access, vol. 7, pp. 102297–102405, 2019.
[10] W. Wu, D. Côté, and Y. Liu, “SAITS: Self-attention-based imputation for time series,” Expert Syst. Appl., vol. 219, Jun. 2023, Art. no. 119619.
[11] A. Ligouri, R. Markovic, M. Ferrando, J. Frisch, F. Casalone, and C. van Treeck, “Augmenting energy time-series for data-efficient imputation of missing values,” Appl. Energy, vol. 334, Mar. 2023, Art. no. 120701.
[12] S. Shan, Y. Li, and J. B. Oliva, “NRTSI: Non-recurrent time series imputation,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Jun. 2023, pp. 1–5.
[13] Y. Tashiro, J. Song, Y. Song, and S. Ermon, “CSDI: Conditional score-based diffusion models for probabilistic time series imputation,” in Proc. Adv. Neural Inf. Process. Syst., vol. 34, 2021, pp. 24804–24816.
[14] I. Silva, G. Moody, D. J. Scott, L. A. Celi, and R. G. Mark, “Predicting in-hospital mortality of ICU patients: The PhysioNet/computing in cardiology challenge 2012,” in Proc. Comput. Cardiol., Sep. 2012, pp. 245–248.
[15] X. Yi, Y. Zheng, J. Zhang, and T. Li, “ST-MVL: Filling missing values in geo-sensor time series data,” in Proc. 25th Int. Joint Conf. Artif. Intell., 2016, pp. 2704–2710.
[16] J. Yoon, J. Jordon, and M. Schaar, “Gain: Missing data imputation using generative adversarial nets,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 5689–5698.
[17] Y. Luo, Y. Zhang, X. Cai, and X. Yuan, “E2GAN: End-to-end generative adversarial network for multivariate time series imputation,” in Proc. 28th Int. Joint Conf. Artif. Intell. Palo Alto, CA, USA: AAAI Press, Aug. 2019, pp. 3094–3100.
