DEEP LEARNING FOR SIMULTANEOUS IMPUTATION AND CLASSIFICATION OF TIME SERIES INCOMPLETE DATA

Chi Cong Nguyen[1], *Cao Truong Tran*[1*], *Bao Ngoc Vi*[1]
DOI: 10.56651/lqdtu.jst.v12.n1.661.ict

Abstract

Classification with time series data has many practical applications in finance, medicine, manufacturing, energy, transportation, and agriculture. However, time series data is often plagued by missing values, which not only decreases the accuracy but also increases the complexity of classification models. A common approach to address classification with missing data is to use imputation methods to estimate the missing values before constructing classification models. Recurrent neural network models have been shown to be effective in estimating missing values for time series data. On the other hand, convolutional neural network models have demonstrated their effectiveness in building classification models for time series data. However, there has been no research that specifically addresses how to combine these two types of models to simultaneously address the problem of time series classification with incomplete data. Therefore, in this paper, a model combining recurrent neural networks and convolutional neural networks is proposed to build a model that can simultaneously estimate missing values and classify time series data. The experimental results demonstrate that the proposed model performs better than the existing methods for time series classification with incomplete data.

Index terms

Time series, classification, imputation, long short-term memory, convolution neural network.

1. Introduction

Classification is a key task in machine learning that involves assigning input data to predefined categories or classes [1]. The goal of classification is to develop a model that can accurately predict the class label of new, unseen data. There are many types of applications of classification, ranging from image recognition to natural language processing to recommendation systems. In image recognition, classification is used to identify objects, such as cars, people, and animals, in images and videos. Image classification is particularly useful in fields such as surveillance, self-driving cars, and

[1]Institute of Information and Communication Technology, Le Quy Don Technical University
[*]Corresponding author, email: truongct@lqdtu.edu.vn

medical imaging [2]. In natural language processing, classification is used to classify documents, such as news articles and emails, into different topics or categories. This is helpful in fields such as journalism, marketing, and information retrieval [3]. In recommendation systems, classification is used to make personalized recommendations to users based on their preferences and behavior. This is often used in e-commerce, streaming services, and social media platforms [4].

Time series data is a type of data that is collected over time and is used to analyze trends and patterns [5]. It is widely used in various fields such as finance, economics, engineering, and social sciences. Time series data can be univariate, which means it consists of a single variable over time, or multivariate, which means it consists of multiple variables over time. The data can be collected at different intervals, such as hourly, daily, weekly, or monthly. Time series data is characterized by autocorrelation, which means that each observation is correlated with its previous observations. Additionally, time series data can exhibit trends, seasonal patterns, and cyclic patterns. Time series analysis techniques are used to identify these patterns and make forecasts based on historical data [6].

Over the past twenty years, time series classification (TSC) has been widely regarded as one of the most difficult challenges in data mining. Since 2015, the availability of temporal data has led to the development of hundreds of TSC algorithms [7]. Time series data, which naturally follows a temporal order, is ubiquitous in tasks requiring human cognitive processing. In fact, any classification problem that involves data registered in a particular order can be viewed as a TSC problem. Time series data appears in many real-world applications, such as electronic health records, human activity recognition, acoustic scene classification, and cyber-security. Furthermore, the UCR/UEA archive, the largest repository of time series datasets, contains a variety of dataset types that demonstrate the diverse applications of the TSC problem [8].

Missing values are very commonly found in time series data due to several factors inherent to the nature of such data [9]. One reason is the presence of measurement errors, which can arise from equipment malfunctions, data collection issues, or human errors. Additionally, sensor failures can occur, leading to periods of missing data when the sensors are not functioning properly. Irregular sampling intervals, where data is collected at varying time intervals, can also contribute to missing values. Data transmission problems during the transfer of data from one system to another can introduce missing values as well. Furthermore, considerations related to data privacy and confidentiality may result in intentional omission of certain data points. Lastly, seasonal or periodic factors can lead to missing values when specific time periods are irrelevant or excluded from the dataset [9], [10].

Missing values in time series classification can pose significant challenges and problems for accurate analysis [9]. Firstly, missing values can disrupt the continuity and temporal dependencies within the data, making it difficult to capture the underlying patterns and dynamics. Time series classification algorithms often rely on these temporal relationships to make accurate predictions. When missing values are present, it becomes challenging to effectively model and interpret the sequential nature of the data. Additionally, missing values can lead to biased or incomplete feature representations, potentially distorting the learned patterns and affecting the classification performance. Furthermore, missing values can result in information loss, reducing the available training data and potentially reducing the overall predictive power of the model [9], [11].

The most common approach to time series classification is to use imputation methods to estimate missing values and create a complete dataset before constructing classification models [9]. Recurrent neural networks (RNNs) have demonstrated superior performance in estimating missing values for time series data [12], [13]. On the other hand, convolutional neural networks (CNNs) have been shown to excel in constructing classifiers for time series data in time series classification tasks [7]. However, there has been no research studying the integration of these two models to build classifiers for time series data containing missing values. Therefore, this paper proposes a hybrid method that combines RNNs and CNNs to construct a classifier capable of simultaneously estimating missing values and classifying the corresponding records.

The rest of this paper is organized as follows. Section 2 gives some related work. Section 3 presents the proposed method. Section 4 shows experimental design. Section 5 shows the results and discussion. Section 6 concludes and states the future work.

2. Related work

2.1. Imputation for time series data

There are various imputation methods available for handling missing values in time series data. These methods aim to estimate the missing values based on the observed data, enabling the construction of complete time series datasets for further analysis [9]. Here are some common imputation methods for time series with missing values:

- *Forward Fill (or Last Observation Carried Forward)*: This method fills missing values with the last observed value. It assumes that the most recent value is a reasonable estimate for the missing data points. This method works well when the time series has a relatively stable trend [14], [15].
- *Backward Fill (or Next Observation Carried Backward)*: Similar to forward fill, this method fills missing values with the next observed value. It assumes that future values are similar to the most recent observation. Backward fill is suitable when the time series exhibits a stable or slowly changing pattern [14], [15].
- *Mean Imputation*: In this method, missing values are replaced with the mean of the available data. It assumes that the mean value represents the central tendency of the time series. Mean imputation is simple to implement but may not capture the temporal dynamics accurately [14], [15].
- *Linear Interpolation*: Linear interpolation estimates missing values based on the linear relationship between adjacent observed values. It assumes a linear trend

between consecutive data points. Linear interpolation is a straightforward method, but it may not be suitable for time series with complex patterns [14], [15].

- *Seasonal Decomposition*: This method decomposes the time series into its seasonal, trend, and residual components. The missing values are then estimated based on the patterns observed in these components. Seasonal decomposition imputation is particularly effective for time series with clear seasonal patterns [14], [15].
- *Time Series Models*: Advanced imputation techniques involve using time series models, such as ARIMA (Autoregressive Integrated Moving Average) or state space models, to estimate missing values. These models take into account the temporal dependencies and patterns in the time series data for more accurate imputation [14], [15].

It is important to note that the choice of imputation method depends on the specific characteristics of the time series, the extent of missingness, and the analysis goals. Evaluating the imputation performance and considering the potential impact on subsequent analyses are crucial steps in selecting the appropriate imputation method for a given time series dataset [9].

2.2. Recurrent neural networks for time series imputation

In the past few decades, various methods have been developed to handle missing values in time series data [14], [15]. However, these approaches suffer from several limitations. They often work only with small amounts of missing data, assume complete randomness in the data, or struggle to handle time series of different lengths. Recently, a new class of methods based on RNNs has emerged as a promising solution [9]. These RNN-based approaches learn to generate new samples that adhere to the temporal patterns found in the training dataset and then use this knowledge to fill in the missing values. These methods, such as GRU-D [12], and BRITS [13] have achieved impressive results in addressing missing values in time series data. One of the key advantages of these models is their strong predictive performance, enabling accurate estimation of missing values. Additionally, they excel at capturing long-term dependencies in the time series and can handle variable-length observations effectively. By leveraging the capabilities of customized RNN architectures and exploiting the temporal relationships between data points, these methods successfully handle the task of filling in missing values in time series datasets.

GRU-D, introduced by [12], is an early attempt at imputing missing values in time series using deep learning models. This research is significant as it recognizes the ability of RNNs to model multivariable time series by leveraging the informative nature of the time dimension. Previous studies, such as [16], attempted to impute missing values using RNNs by concatenating timestamps and raw data, treating timestamps as an additional attribute. However, [12] introduces the concept of time lag. In this paper, the authors employ the Gated Recurrent Unit (GRU) to generate missing values. In each layer of the GRU, they handle missing inputs by replacing them with a combination of existing values. The primary contribution of this research is the GRU-D model, which integrates the GRU architecture with the imputation task, and the introduction of the decay rate. The authors discover that when addressing missing value imputation, variables that have been missing for a prolonged period tend to be close to a default value. Furthermore, they observe that the influence of input variables diminishes over time if a variable has been missing for an extended duration.

Unlike previous methods, BRITS [13] takes a completely RNN-based approach and proposes a imputation model using unidirectional dynamics. To handle irregular time series data, BRITS introduces the concept of time lag. In a similar vein to the decay rate concept in GRU-D, BRITS suggests temporal decay. While GRU-D incorporates time lags as part of the input and utilizes them as the decay rate, BRITS updates the hidden states with the decay rate. This means that when updating the hidden state, the previously hidden state decays based on the recorded time duration in the time lag. As a bidirectional version, BRITS employs a bidirectional RNN [17], [18] by utilizing both the forward and backward recurrent dynamics. This entails training two models, one in the forward direction and the other in the backward direction. To ensure consistency between the two directions, BRITS introduces a consistency loss that considers the losses from both directions.

2.3. Classification with time series data

Traditional machine learning and deep learning are two different approaches to time series classification, each with its strengths and limitations [7], [19]. In traditional machine learning, feature engineering plays a crucial role. Domain experts manually extract relevant features from the time series data, which are then used as inputs to the classification algorithms. Traditional algorithms like decision trees, support vector machines (SVM), and random forests are commonly applied. These methods are interpretable, computationally efficient, and can work well with small to medium-sized datasets. However, they often struggle to capture complex temporal patterns and dependencies in time series data, especially when dealing with high-dimensional or long-term sequential information [7].

On the other hand, deep learning approaches have revolutionized time series classification. Deep learning models can automatically learn hierarchical representations and abstract features directly from the raw time series data. This eliminates the need for manual feature engineering and enables the capture of intricate temporal dependencies. Deep learning models excel in handling large-scale datasets, high-dimensional input, and long-term sequential information. However, they generally require more computational resources, larger amounts of labeled data, and can be less interpretable compared to traditional machine learning algorithms [7], [8].

The choice between traditional machine learning and deep learning for time series classification depends on various factors, such as the complexity of the problem, available computational resources, size of the dataset, and the interpretability requirements. While traditional machine learning can be effective in certain scenarios, deep learning approaches have demonstrated superior performance in many challenging time series classification tasks, especially when dealing with complex and large-scale datasets [7], [19].

Convolutional Neural Networks (CNNs) have been widely adopted for time series classification tasks due to their remarkable ability to capture spatial dependencies and extract meaningful features [7], [19]. CNNs leverage the concept of convolutions to analyze local patterns and relationships within the time series data. In the context of time series classification, the input is treated as a one-dimensional signal, and the CNN applies filters over small windows, sliding across the time series to extract relevant features. The convolutional layers help to detect patterns and temporal dependencies at different scales, allowing the network to learn representations that are invariant to small temporal shifts. Subsequent pooling layers are often used to reduce the spatial dimensionality and retain the most informative features. Additionally, CNNs for time series classification often include fully connected layers and a softmax output layer for accurate class predictions. The combination of convolutional and pooling layers, along with the ability to learn hierarchical representations, makes CNNs effective in capturing both local and global temporal features, leading to improved classification performance with time series data [7], [19].

3. The proposed method

In this section, we present the proposed GANRES method for simultaneously performing imputation and classification for time series data. The core idea of the proposed method is to utilize a generative adversarial network (GAN) with a recurrent network as its core to estimate the missing values in the time series data. Once the missing values have been estimated, the output of the GAN model is then classified using a CNN model. This proposed method combines the strengths of RNN and GAN models in estimating missing values and the CNN model in classifying time series data. Integrating these two models into a single framework enables the model to effectively estimate missing values and accurately classify time series data.

The overall architecture of GANRES is depicted in Fig.1. GANRES takes as input the incomplete time series data, data-label pairs, mask matrix, and time-lag matrix. The framework consists of three key components: a generator G, a discriminator D, and a classifier C. Utilizing the available information in the time series data, the generator G predicts the missing components, generating an imputed time series matrix to fool the discriminator D. Additionally, the imputed time series is used to train the classifier C. The classifier C provides feedback to the generator G through the cross entropy loss term. During the imputation process of an incomplete sample, the classifier guides the generator to focus on time series samples with the same label.

3.1. Generator network

The objective of the generator is to learn the distribution of the observed multivariate time series data and estimate the missing values in the input time series. The generator

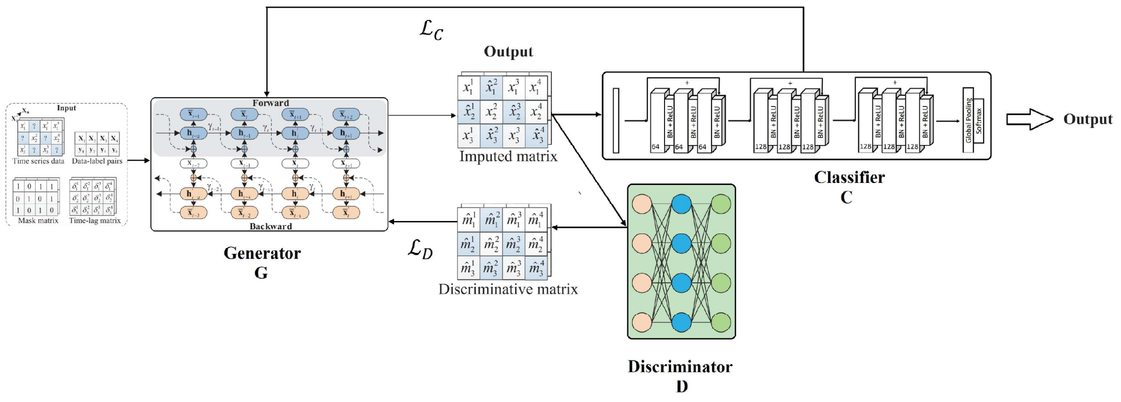

Fig. 1. The architecture of GANRES.

takes time series samples X, mask matrix M, and time-lag matrix δ as inputs and generates the imputed time series data $\widehat{X}$ as output. A key component of the generator network structure is the BiRNN cell. The BiRNN cell incorporates a time decay vector to capture important information from past and future steps in a multivariable time series. The cell takes a time series sample, along with forward and backward time lag matrices, as inputs. It generates reconstructed time series for both forward and backward directions. The time decay vector controls the influence of previous observations by incorporating a decay factor. A fully connected layer is used to reconstruct the time series based on the previous hidden state. Missing values in the input are replaced with corresponding values in the reconstruction. The next hidden state is computed using the time decay vector and the imputed vector. The forward and backward hidden states and reconstructed time series are obtained in both directions.

To account for characteristic correlations in the data, a feature-based estimate is introduced. This estimate incorporates information from other characteristics at the same time step. A separate observation is obtained first, and a feature-based estimate is computed based on this observation. The weight assigned to combine the history-based estimate and the feature-based estimate is learned using the temporal decay and masking vector. The missing values in the input are replaced with values from the combined vector. The resulting vector is passed to the next recurrent stage to forecast the memory. The estimation loss is calculated as the sum of the errors between the actual values and the reconstructed values for the different estimates.

The generator's objective includes three types of losses: the adversarial loss, the classifier loss, and the reconstruction loss. The adversarial loss is similar to that of regular GANs. The classifier loss represents the feedback from the classifier, which will be specified later. The reconstruction loss is used to ensure the consistency between the observed and reconstructed time series. We denote the forward and backward reconstructed time series as $\overline{X}$ and $\overline{X}'$, respectively. The reconstruction losses $\mathcal{L}(X, \overline{X})$ and $\mathcal{L}(X, \overline{X}')$ are defined as follows:

$$\mathcal{L}(X, \overline{X}) = M \odot l_e(X, \overline{X}) \tag{1}$$

$$\mathcal{L}(X, \overline{X}') = M \odot l_e(X, \overline{X}') \tag{2}$$

Here, l_e represents the mean absolute error function. Additionally, we compute the reconstruction error between $\overline{X}$ and $\overline{X}'$ to ensure consistency in both directions:

$$\mathcal{L}(\overline{X}, \overline{X}') = M \odot l_e(\overline{X}, \overline{X}') \tag{3}$$

3.2. Discriminator network

Following the typical GAN methodology, we use a discriminator to compete with the generator, which can aid the generator in producing as accurate data as feasible. Unlike the traditional discriminator, which produces a scalar value, the discriminator in GANRES produces a probability matrix, with each value representing the authenticity degree of each component in the imputed time series $\widehat{X}$.

The discriminator attempts to discriminate between the observed and created components of the time series $\widehat{X}$ imputed by the generator. Inspired by [20], we add a temporal reminder matrix R as an extra input to the discriminator. A portion of the missing information in a time series is represented by the temporal reminder matrix R.

It can motivate the generator to correctly understand the generator's unique real data distribution. It is more specifically characterized as:

$$R = K \odot M + 0.5(1 - K) \tag{4}$$

where the components in $K = (K_1, ..., K_n) \in \{0, 1\}^{d \times n}$ are randomly set to 1 or 0 in each training epoch. In other words, R discloses certain components of M to the discriminator (i.e., their states in K are set to 1).

As a consequence, the discriminator D takes as input the imputed time series $\widehat{X}$ and the temporal reminder matrix R. The fully linked layer is then used to calculate the likelihood that each component of the time series will be detected. The forward discriminative probability, represented by $\overline{m}_t$, is computed as $\overline{m}_t = W_x^D h_t^D + b_x^D$, where W_x^D and b_x^D are discriminator parameters that must be learned. As a result, the discriminator produces a forward discriminative matrix, which is stored in $\overline{M} = (\overline{m}_1, ..., \overline{m}_n) \in R^{d \times n}$. Similarly, we get a backward discriminative matrix $\overline{M}' = (\overline{m}'_1, ..., \overline{m}'_n) \in R^{d \times n}$. As a result, the discriminator's output, indicated by $D(\widehat{X}, R)$, is expressed as:

$$D(\widehat{X}, R) = \widehat{M} = \frac{\overline{M} + \overline{M}'}{2} = (\widehat{m}_1, ..., \widehat{m}_n) \tag{5}$$

The loss function $\mathcal{L}_D$ in the discriminator, which is the probability of correctly predicting the mask matrix M, is defined as:

$$\mathcal{L}_D = -\mathbb{E}\left[M \odot log\widehat{M} + (1 - M) \odot log(1 - \widehat{M})\right] \tag{6}$$

3.3. Classifier network

To improve the model's accuracy, we propose incorporating a classifier specifically designed for time series data. In [7], the authors conducted an extensive empirical analysis of deep neural networks (DNNs) for TSC. Their work highlighted the success of deep learning approaches in various domains and implemented nine state-of-the-art deep learning classifiers. The results demonstrated that deep learning models, particularly using the Residual Network architecture, achieved impressive performance for TSC.

The RESNET architecture, introduced in [21], plays a crucial role in our classifier. The output of the classifier, denoted as $C(\widehat{X})$, can be expressed as follows:

$$C(\widehat{X}) = \text{RESNET}(\widehat{X}) = \widehat{y} \tag{7}$$

Consequently, the loss function for the classifier, denoted as $\mathcal{L}_C$, is defined as the binary cross-entropy loss between the true labels y and the predicted labels $\widehat{y}$:

$$\mathcal{L}_C(y, \widehat{y}) = \mathcal{L}_{\text{CE}}(y, C(\widehat{X})) \tag{8}$$

4. Experiment design

4.1. Datasets and benchmark methods

The proposed method is evaluated on three time series datasets, as presented in Table 1. These three datasets are PhysioNet, Electrocardiogram and Wafer.

PhysioNet Challenge 2012 (PhysioNet) [22] is a publicly available dataset which consists of multivariate clinical time series data from 8,000 intensive care unit (ICU) records. Each record represents a multivariate time series spanning approximately 48 hours and includes 35 properties such as albumin, heart rate and glucose. It is important to note that this dataset has a high sparsity with a total of 78% missing values [22].

Electrocardiogram (ECG) data has been utilized to investigate heartbeat classification using deep neural network architectures, as well as to explore the potential of transfer learning in this context. The dataset consists of ECG signals representing the shapes of heartbeats under different conditions, including normal cases, various arrhythmias, and myocardial infarction. Prior to analysis, the signals are subjected to preprocessing and segmentation, where each segmented portion corresponds to a single heartbeat [9].

Wafer data pertains to the fabrication of semi-conductor microelectronics. It comprises a set of inline process control measurements obtained from different sensors throughout the processing of silicon wafers for semiconductor manufacturing. The Wafer database consists of multiple datasets, each captures the measurements recorded by a specific sensor during the processing of an individual wafer using a particular tool. The datasets are categorized into two classes: normal and abnormal, reflecting different operating conditions [9].

Table 1. Datasets used in experiments

Dataset	**Time slat**	**Features**	**Train**	**Validate**	**Test**
PhysioNet	48	35	2400	800	800
ECG	152	2	80	20	100
Wafer	198	6	238	60	896

In our experiments, we evaluate various interpolation and imputation methods alongside other prediction baselines. Two state-of-the-art methods, GRU-D [12] and BRITS [13], utilize bidirectional recurrent networks for time series imputation. A variant of BRITS, called BRITS-RESNET, incorporates BRITS for imputing missing data, followed by feeding the complete data into a RESNET model for classification. Once the data is imputed, it is inputted into the RESNET model for classification. Imputation methods like GRU-D and BRITS directly classify the data using the sigmoid function. Additionally, we conduct a comparative test by applying the BRITS model for data imputation and using a RESNET model for classification, providing a benchmark for comparison with our proposed model.

4.2. Parameter settings

During the implementation, the deep learning baselines are configured as in [12]. The learning rate, the dropout rate, the number of hidden units, the number of training epochs and the batch size are 0.001, 0.3, 64, 100 and 64, respectively. The Adam algorithm is used for network training. The generator is trained once every five optimization steps of the discriminator, while the classifier is trained once every five optimization steps of the generator. The hyperparameters α and β of the generator are both set to 5. The reminder rate, which determines how many missing states in matrix M are encoded in the temporal reminder matrix for the discriminator, is set to 0.8. The label rate is 100%. To assess the imputation performance, we utilize the mean absolute error (MAE) and mean relative error (MRE).

5. Results and discussion

5.1. Performance comparison for time series imputation

During the training phase for the imputation task, we randomly remove 10% to 80% of observed values from the ECG and Wafer datasets. These removed values are used as the ground truth for calculating the MAE and MRE values. Figs 2 and 3 provide a visual comparison between several modern imputation methods and the proposed method. Fig. 2 specifically focuses on the performance comparison (in terms of MAE) of time series imputation methods at different missing rates using the ECG dataset. Similarly, Fig. 3 illustrates the performance comparison using the Wafer dataset.

The comparison of different imputation methods, as shown in Figs. 2 and 3, indicates that the GRU-D method is less effective compared to other methods. The reason behind this is that GRU-D utilizes a one-dimensional RNN model, which fails to fully capture the bidirectional correlations present in the time series data. However, our proposed BRITS and GANRES models improve upon the GRU-D approach by employing a bidirectional RNN model, which comprehensively captures the correlations between the values in the time series. The results demonstrate that our proposed method achieves a lower MAE than BRITS, especially when dealing with a high missing rate in the data. Furthermore, by incorporating a discriminator network within the GAN framework, our GANRES model can effectively distinguish between generated values and imputed values. This leads to improved performance compared to the BRITS model.

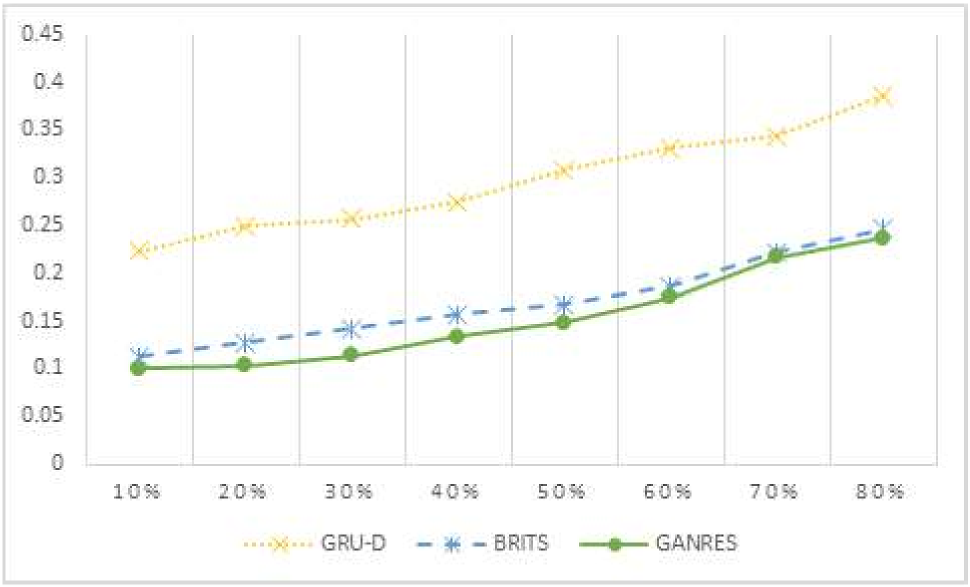

Fig. 2. Performance comparison (in MAE) of time series imputation methods under different missing rates with ECG dataset.

Table 2 presents a comparison of the MRE measurements on the ECG and Wafer datasets at different levels of missing values. It is evident from the table that our proposed method achieves smaller MRE values compared to GRU-D and BRITS, further confirming its superiority.

In summary, the comparison of imputation methods highlights the superior performance of the proposed method over GRU-D, as it effectively captures bi-directional correlations in time series data. Additionally, the incorporation of a discriminator network in the GANRES model further enhances its performance compared to the BRITS model, leading to improved imputation results.

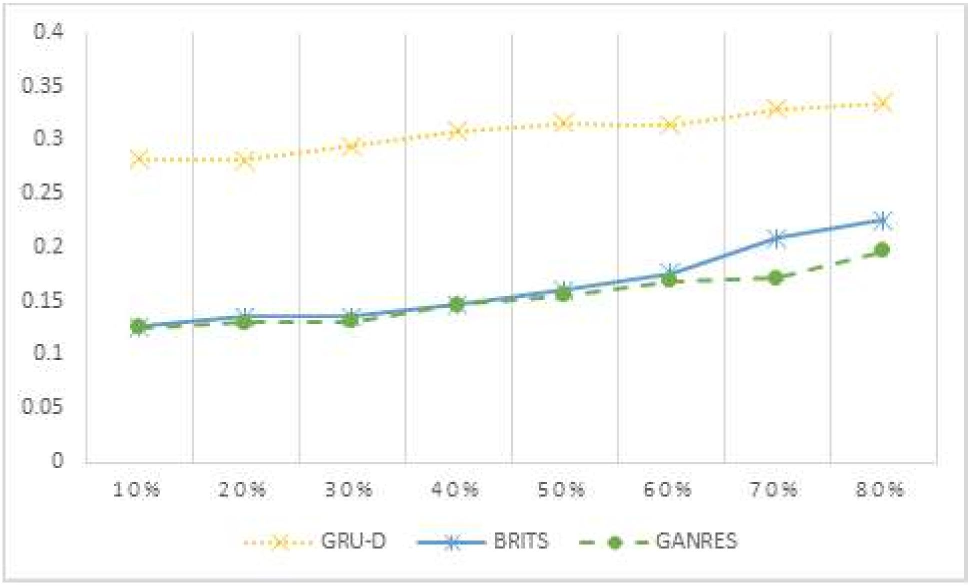

Fig. 3. Performance comparison (in MAE) of time series imputation methods under different missing rates with Wafer dataset.

Table 2. MRE performance comparison of time series imputation methods under different missing rates

Missing rate	**Dataset**					
	ECG			Wafer		
	GRU-D	BRITS	GANRES	GRU-D	BRITS	GANRES
10%	0.672	0.458	**0.445**	0.303	0.126	0.127
20%	0.685	0.461	**0.459**	0.323	0.135	**0.129**
30%	0.695	0.465	**0.462**	0.331	0.134	**0.133**
40%	0.702	0.472	**0.470**	0.365	0.150	**0.139**
50%	0.714	0.479	**0.478**	0.379	0.157	**0.152**
60%	0.742	0.486	**0.481**	0.385	0.186	**0.165**
70%	0.753	0.499	**0.483**	0.394	0.219	**0.158**
80%	0.795	0.523	**0.512**	0.402	0.236	**0.183**

5.2. Performance comparison for time series classification

In Table 3, the classification accuracy of the proposed method and other approaches is presented. The results in Table 3 indicate that imputation models like GRU-D and BRITS perform poorly when applied to time series datasets for classification. However, by incorporating the RESNET classification model, significant improvements in classification outcomes are observed. Experimental results demonstrate that employing the BRITS imputation model followed by the RESNET classification model produces better results for time series datasets. The proposed GANRES model capitalizes on the data correlation within the data generator and leverages the strengths of the RESNET model for data classification, leading to superior performance compared to other models.

Table 3. Performance Comparison (in AUC) for Classification Tasks with external time series assignment methods

Model	**Dataset**						
	Physionet	ECG			Wafer		
	Missing rate						
	78%	30%	50%	70%	30%	50%	70
GRU-D	80.35 (±1.52)	70.45 (±2.11)	65.73 (±1.28)	64.55 (±6.07)	62.76 (±2.59)	54.53 (±2.62)	52.45 (±1.49)
BRITS	82.15 (±0.90)	70.10 (±3.67)	66.76 (±1.69)	65.85 ±6.56)	63.41 (±2.95)	57.97 (±0.65)	53.26 (±1.13)
BRITS-RESNET	80.84 (±0.77)	81.41 (±1.74)	82.45 (±0.96)	77.39 (±1.75)	73.45 (±2.50)	73.45 (±1.89)	67.61 (±2.71)
GANRES	**84.58** (±0.56)	**85.84** (±0.73)	**83.27** (±1.35)	**87.29** (±0.88)	**92.80** (±6.17)	**89.21** (±1.89)	**88.34** (±1.39)

Fig. 4 depicts the ineffectiveness of external imputation methods for the Physionet dataset, primarily attributed to its large number of attributes and high missing rate. However, the GRU-D and BRITS models exhibit commendable performance on this dataset, as previously evidenced by the authors [12], [13]. The GANRES model extends the foundation of the BRITS model for data generation, incorporating a discriminator network that enables the generated data to closely resemble real data. The loss functions of the classification network in GANRES further augment the data generation process within the generative model.

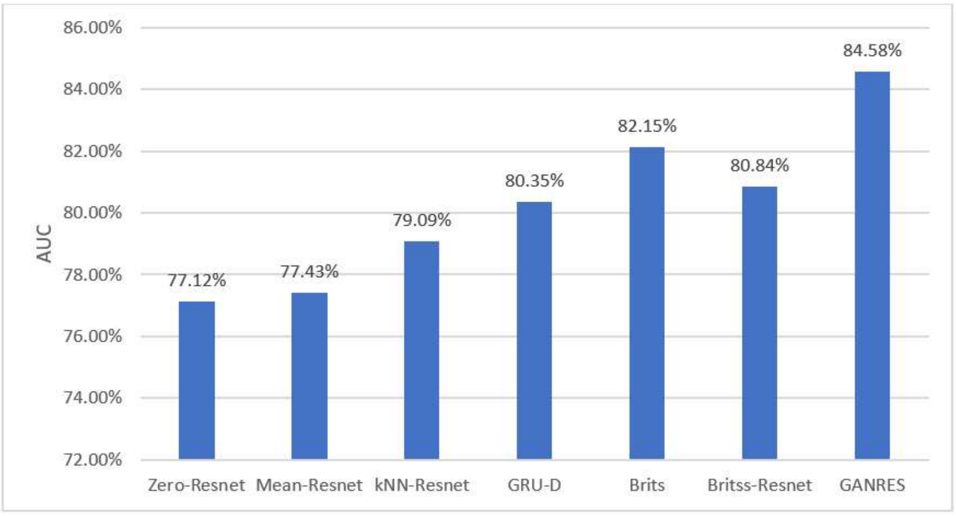

Fig. 4. Classification performance on Physionet datasets with different imputation methods.

In summary, the experimental results demonstrate that the proposed GANRES model, which combines the BRITS imputation model with the RESNET classification model, outperforms other methods in terms of classification accuracy for time series datasets. The GANRES model effectively leverages data correlation and the strengths of both models, leading to superior performance in data classification tasks.

6. Conclusion

In conclusion, classification with time series data is crucial in various industries, but missing values pose significant challenges. Imputation methods are commonly used to estimate missing values before constructing classification models, with recurrent neural network models being effective in this regard. Meanwhile, convolutional neural network models excel in building classification models for time series data. However, the combination of these two models to address time series classification with incomplete data has not been extensively explored. This paper has introduced a novel model that integrates recurrent and convolutional neural networks to simultaneously estimate missing values and classify time series data. Experimental results demonstrate the superiority of the proposed model compared to existing methods in handling time series classification with incomplete data. This research fills a gap in the field and provides a promising approach to enhance the accuracy of classification models for time series data with missing values.

References

[1] J. Han, J. Pei, and H. Tong, *Data mining: concepts and techniques*. Morgan Kaufmann, 2022.

[2] A. Singh and P. Singh, "Image classification: a survey," *Journal of Informatics Electrical and Electronics Engineering (JIEEE)*, vol. 1, no. 2, pp. 1–9, 2020. doi: 10.54060/JIEEE/001.02.002

[3] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, "Text classification algorithms: A survey," *Information*, vol. 10, no. 4, p. 150, 2019. doi: 10.3390/info10040150

[4] D. Jannach, A. Manzoor, W. Cai, and L. Chen, "A survey on conversational recommender systems," *ACM Computing Surveys (CSUR)*, vol. 54, no. 5, pp. 1–36, 2021. doi: 10.1145/3453154

[5] J. D. Hamilton, *Time series analysis*. Princeton university press, 2020.

[6] R. H. Shumway, D. S. Stoffer, and D. S. Stoffer, *Time series analysis and its applications*. Springer, 2000, vol. 3.

[7] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data mining and knowledge discovery*, vol. 33, no. 4, pp. 917–963, 2019. doi: 10.1007/s10618-019-00619-1

[8] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *2017 International joint conference on neural networks (IJCNN)*. IEEE, 2017. doi: 10.1109/IJCNN.2017.7966039 pp. 1578–1585.

[9] C. Fang and C. Wang, "Time series data imputation: A survey on deep learning approaches," *arXiv preprint arXiv:2011.11347*, 2020.

[10] F. M. Bianchi, L. Livi, K. Ø. Mikalsen, M. Kampffmeyer, and R. Jenssen, "Learning representations of multivariate time series with missing data," *Pattern Recognition*, vol. 96, p. 106973, 2019. doi: 10.1016/J.PATCOG.2019.106973

[11] Z. C. Lipton, D. C. Kale, R. Wetzel *et al.*, "Modeling missing data in clinical time series with RNNs," *Machine Learning for Healthcare*, vol. 56, pp. 253–270, 2016.

[12] 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. doi: 10.1038/s41598-018-24271-9

[13] W. Cao, D. Wang, J. Li, H. Zhou, L. Li, and Y. Li, "Brits: Bidirectional recurrent imputation for time series," *Advances in neural information processing systems*, vol. 31, 2018.

[14] S. Moritz, A. Sardá, T. Bartz-Beielstein, M. Zaefferer, and J. Stork, "Comparison of different methods for univariate time series imputation in R," *arXiv preprint arXiv:1510.03924*, 2015.

[15] S. Moritz and T. Bartz-Beielstein, "imputeTS: Time series missing value imputation in R," *R J.*, vol. 9, no. 1, p. 207, 2017. doi: 10.32614/RJ-2017-009

[16] Z. C. Lipton, D. Kale, and R. Wetzel, “Directly modeling missing data in sequences with rnns: Improved classification of clinical time series,” in *Machine learning for healthcare conference*. PMLR, 2016, pp. 253–270.
[17] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997. doi: 10.1109/78.650093
[18] M. Berglund, T. Raiko, M. Honkala, L. Kärkkäinen, A. Vetek, and J. T. Karhunen, “Bidirectional recurrent neural networks as generative models,” *Advances in neural information processing systems*, vol. 28, 2015.
[19] A. Gupta, H. P. Gupta, B. Biswas, and T. Dutta, “Approaches and applications of early classification of time series: A review,” *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 1, pp. 47–61, 2020. doi: 10.1109/TAI.2020.3027279
[20] X. Miao, Y. Wu, J. Wang, Y. Gao, X. Mao, and J. Yin, “Generative semi-supervised learning for multivariate time series imputation,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 10, 2021. doi: 10.1609/aaai.v35i10.17086 pp. 8983–8991.
[21] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016. doi: 10.1109/cvpr.2016.90 pp. 770–778.
[22] 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 *2012 Computing in Cardiology*. IEEE, 2012, pp. 245–248.

Manuscript received 21-04-2023; Accepted 12-06-2023.

■

Chi Cong Nguyen received the master degree in computer science in 2022 from Le Quy Don Technical University. Currently, he is a lecturer at the Institute of Information and Communication Technology - Le Quy Don Technical University. He is working in machine learning, specializing in machine learning for data mining with missing data. Main research directions include natural language processing, intelligent computing and deep learning.
Email: congnc@lqdtu.edu.vn

Cao Truong Tran received the PhD degree in computer science from Victoria University of Wellington, New Zealand. He also did postdoc at Victoria University of Wellington. He is researching in the field of machine learning and evolutionary computation, specialized with evolutionary machine learning for data mining with missing data. He servers as a reviewer of international journals, including IEEE Transactions on Evolutionary Computation, IEEE Transactions on Cybernetics, Pattern Recognition, Knowledge-Based Systems, Applied Soft Computing and Engineering Application of Artificial Intelligence. He is also a PC member of international conferences, including IEEE Congress on Evolutionary Computation, IEEE Symposium Series on Computational Intelligence, the Australasian Joint Conference on Artificial Intelligence and the AAAI Conference on Artificial Intelligence.
Email: truongct@lqdtu.edu.vn

Bao Ngoc Vi received the master degree in computer science in 2010 from Le Quy Don Technical University. She has been working as a lecturer at the Institute of Information and Communication Technology - Le Quy Don Technical University. She is researching in the field of machine learning, natural language processing.
Email: ngocvb@lqdtu.edu.vn

HỌC SÂU CHO ĐỒNG THỜI ƯỚC LƯỢNG VÀ PHÂN LỚP DỮ LIỆU CHUỖI THỜI GIAN BỊ THIẾU GIÁ TRỊ

Nguyễn Chí Công, Trần Cao Trưởng, Vi Bảo Ngọc

Tóm tắt

Dữ liệu bị thiếu giá trị là một vấn đề phổ biến gây khó khăn trong việc xây dựng các mô hình phân loại dữ liệu chuỗi thời gian. Bài báo tập trung giải quyết bài toán phân loại dữ liệu chuỗi thời gian có giá trị thiếu. Để khắc phục điều này, bài báo đề xuất mô hình mới kết hợp cả mạng nơ-ron hồi qui (RNN) và mạng nơ-ron tích chập (CNN) để đồng thời ước lượng giá trị thiếu và phân loại dữ liệu chuỗi thời gian. Thông qua việc tích hợp RNN và CNN, mô hình đề xuất có khả năng đạt hiệu suất tốt hơn so với các phương pháp hiện có trong phân lớp dữ liệu chuỗi thời gian có giá trị thiếu. Qua việc thử nghiệm, đánh giá trên các tập dữ liệu thực tế, kết quả cho thấy mô hình đề xuất có khả năng phân loại chính xác hơn dữ liệu chuỗi thời gian có giá trị thiếu. Nghiên cứu này cung cấp một hướng tiếp cận tiềm năng để nâng cao hiệu suất của các mô hình phân loại dữ liệu chuỗi thời gian bị thiếu giá trị.

Từ khóa

Dữ liệu bị thiếu, dữ liệu chuỗi thời gian, học sâu, mô hình hồi qui, mô hình tích chập.