DAGAN:Generative Adversarial Network with Dual Attention-Enhanced GRU for Multivariate Time Series Imputation

Hongtao Song, Xiangran Fang, Dan Lu(✉), and Qilong Han

College of Computer Science and Technology,
Harbin Engineering University, Harbin 150000, China
{songhongtao,fxran,ludan,hanqilong}@hrbeu.edu.cn

Abstract. Missing values are common in multivariate time series data, which limits the usability of the data and impedes further analysis. Thus, it is imperative to impute missing values in time series data. However, in handling missing values, existing imputation techniques fail to take full advantage of the time-related data and have limitations in capturing potential correlations between variables. This paper presents a new model for imputing multivariate time series data called DAGAN, which comprises a generator and a discriminator. Specifically, the generator incorporates a Temporal Attention layer, a Relevance Attention layer, and a Feature Aggregation layer. The Temporal Attention layer utilizes an attention mechanism and recurrent neural network to address the RNN's inability to model long-term dependencies in the time series. The Relevance Attention layer employs a self-attention-based network architecture to capture correlations among multiple variables in the time series. The Feature Aggregation layer integrates time information and correlation information using a residual network and a Linear layer for effective imputation of missing data. In the discriminator, we also introduce a temporal cueing matrix to aid in distinguishing between generated and real values. To evaluate the proposed model, we conduct experiments on two real-time series datasets, and the findings indicate that DAGAN outperforms state-of-the-art methods by more than 13%.

Keywords: Time series imputation · Self-Attention · Recurrent Neural Network · Generative Adversarial Network

1 Introduction

Time series data refers to data arranged in chronological order that reflects the state of things as they change over time [25]. Typical time series data includes factory sensor data, stock trading data, climate data for a particular region,

This work was supported by the National Key R&D Program of China under Grant No. 2020YFB1710200 and Heilongjiang Key R&D Program of China under Grant No. GA23A915.

© The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2024
B. Luo et al. (Eds.): ICONIP 2023, CCIS 1966, pp. 258–271, 2024.
https://doi.org/10.1007/978-981-99-8148-9_21

power data, transportation data, and so on. In practical applications, multivariate time series data is more common. However, due to the complexity of real-world conditions, multivariate time series data often contains missing values for various reasons [14,26], such as sensor malfunctions, communication errors, and accidents [13]. At the same time, in reality, due to equipment issues, such as the sampling machine itself not having a fixed sampling rate, even if the dataset is complete, the intervals between the data points are often irregular, which can also be considered a problem of missing data. These missing values undermine the interpretability of the data and can seriously affect the performance of downstream tasks such as classification, prediction, and anomaly detection [12]. To further study this incomplete time series, imputing the missing values is an inevitable step.

In order to solve the problem of missing data, many time series imputation methods have been proposed to infer missing values from observed values. Traditional data imputation methods are based on mathematical and statistical theory and fill in missing values by analyzing the data. However, this method ignores the inherent time correlation imputation in time series data. In contrast, time series imputation methods based on deep learning consider the time information in multivariate time series data and achieve better imputation accuracy. Recurrent neural networks rely on the calculation of hidden states for each unit to capture time dependence, but they still place more emphasis on the outputs of adjacent time steps and have difficulty capturing and utilizing long-term dependencies in time series [9].

In recent years, Generative Adversarial Networks (GANs) have shown outstanding performance in the field of data generation. However, general GANs do not consider the time correlation of time series data and ignore the potential correlation between different features of multivariate time series. Therefore, in this paper, we propose a new multivariate time series imputation model called DAGAN. DAGAN consists of a generator and a discriminator. The generator utilizes a gated recurrent neural network to learn the time information in multivariate time series data. It performs a weighted sum of hidden states in the recurrent neural network using an attention mechanism, which improves the model's ability to learn long-term dependencies in time series while ensuring that the model focuses more on important information. Furthermore, we use a self-attention mechanism to link different variables in multivariate time series, allowing all time steps to participate in each layer to maximize the accuracy of multivariate time series imputation. The task of the discriminator is to distinguish between real and generated values. In this paper, we use a temporal cueing matrix to improve the performance of the discriminator, which contains some missing information from the original time series data. This further forces the generator to learn the real data distribution of the input dataset more accurately.

In summary, compared with existing time series imputation models, the model proposed in this paper has the following contributions:

1) We propose a time series imputation method that combines attention mechanisms and gated recurrent neural networks to enhance the ability of recur-

rent neural networks to utilize the long-term dependencies of time series data, while ensuring that the model focuses on important information and improves the quality of imputation.
2) We use the masked self-attention mechanism to capture relationships between different features in the multivariate time series. By incorporating self-attention at each layer, all time steps participate, maximizing the accuracy of imputation for multivariate time series data.
3) We conduct experiments on two different real datasets, using the root mean square error (RMSE) as the performance indicator. The results show that, in most cases, our model outperforms the baseline method.

2 Related Work

There are two main methods for dealing with missing values in time series analysis: direct deletion and imputation [11]. The main idea behind the direct deletion method is to delete samples or features containing missing values directly. Although this method is simple and easy to implement, it can lead to a reduction in sample size and the loss of important information [10]. The imputation method can be divided into statistical-based, machine-learning-based, and neural network-based methods, based on different imputation techniques and technologies.

Statistical-based imputation methods use simple statistical strategies, such as imputing missing values with mean or median values [11,14]. For example, SimpleMean [7]imputes missing values by calculating the mean value. Although these methods can quickly fill in missing values, they can affect the variance of original data and have poor imputation accuracy.

Common machine-learning-based imputation methods include k-nearest neighbor (KNN) method, expectation-maximization method, and random forest. The basic idea of KNN algorithm is to find the k nearest neighboring data points with known values to a missing data point and then calculate a weighted average by some rule, using the known values of these neighbors, to obtain the imputed value. Random forest is a decision-tree-based imputation method [24]. Although these methods exhibit better imputation performance than the previous two methods, they lack the ability to utilize the time correlation and complex correlation between different variables.

In time series data, variables change over time and are interrelated. Therefore, it is crucial to use the time correlation of time series data to improve imputation accuracy. With the rapid development of neural network technology, researchers have started to apply it to the field of time series data imputation [6,10,18]. To properly handle time series data with missing values, some researchers have proposed using RNN-based methods to process missing values. RNN-based data imputation models can capture the time dependency of time series data [19,22]. Among them, GRUD [3], which is based on gated recurrent neural networks, uses hidden state decay to capture time dependence. BRITS [2] treats missing values as variables and imputes them based on the hidden states of bidirectional RNN. GLIMA [23] uses a global and local neural network model

with multi-direction attention to process missing values, which partly overcomes the problem that RNN heavily depends on output from nearby timestamps. The problem with RNN-based models is that after a period of time, the weight of current inputs becomes negligible, but they should not be ignored [2,28]. However, this issue does not exist in GAN-based models. GANs generate data with the same distribution as the original data through the game between generator and discriminator. Examples of GAN-based imputation models include GRUIGAN [15], GAIN [28], BiGAN [8], and SSGAN [17]. These methods take advantage of the benefits of GANs and combine them with RNN to improve their ability to capture time dependence.

3 Preliminary

Before introducing the model proposed by us, we provide a formal definition of multivariate time-series data and explain some symbols used in certain models.

Given a set of multivariate time series $X = (x_1, x_2 \cdots, x_T), \quad \in \mathbb{R}^{T \times D}$ and timestamps $TS = \{t_1, t_2, \ldots, t_T\}$, the t-th observation $\mathbf{x}_t \in \mathbb{R}^D$ consists of D features $\{x_{t1}, x_{t2}, \ldots, x_{tD}\}$ and corresponds to timestamp t_t. We define a mask matrix M, which is used to indicate the positions of missing values in the dataset. When x_{ij} is missing, the corresponding element in the mask matrix is equal to 0, otherwise, it is 1. The mask matrix M has the same size as the input dataset. The specific formula is shown as follows.

$$m_{ij} = \begin{cases} 0 & \text{if } x_{ij} \text{ is missing} \\ 1 & \text{otherwise} \end{cases} \tag{1}$$

We also introduce the time interval matrix δ, which is represented as follows.

$$\delta_{ij} = \begin{cases} \delta_{ij-1} + t_i - t_{i-1}, & \text{if } m_{ij-1} = 0, i > 0 \\ t_i - t_{i-1}, & \text{if } m_{ij-1} = 1, i > 0 \\ 0, & i = 0 \end{cases} \tag{2}$$

The time retention matrix represents the time interval between the current time and the last valid observation. It is also a matrix of the same size as the input data set. Then, we introduce the time decay factor α, which is used to control the influence of past observations. When the δ value is higher, α becomes smaller. This indicates that the further the missing value is from the last true observation value, the more unreliable its value is.

$$\alpha_i = \exp\left(-\max\left(0, W_\alpha \delta_i + b_\alpha\right)\right) \tag{3}$$

4 Model

4.1 The Overall Structure

In this section, we will introduce the overall architecture of the DAGAN model. Figure 2 shows the overall architecture of DAGAN. The input to DAGAN

includes time-series data, mask matrix, and time interval matrix. DAGAN consists of a generator and a discriminator. The generator generates imputed data based on the observed values in the time-series data. Its objective is to deceive the discriminator with the generated data. The discriminator takes the estimated time-series matrix and the temporal attention matrix as inputs. It attempts to differentiate between the generated data and the real data. Specifically, we introduce the time-attention matrix to encode a part of the missing information in the time-series data stored in the mask matrix. Below, we will describe the structure of each module in detail (Fig. 1).

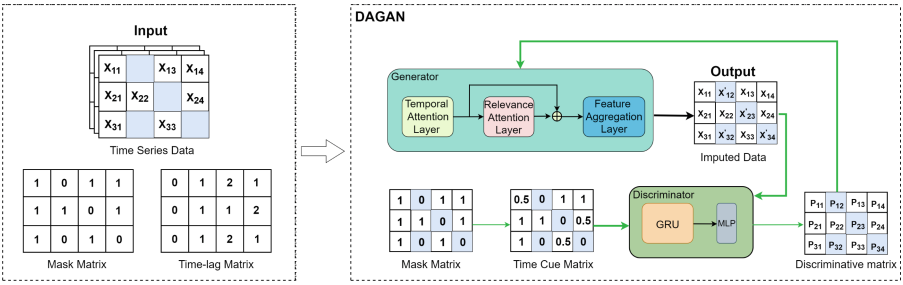


Fig. 1. The overall architecture diagram of DAGAN.

4.2 Generator Network

The goal of the generator is to learn the distribution of multivariate time series data and generate missing values. Our generator includes a Temporal Attention layer, a Relevance Attention layer, and a Feature Aggregation layer. The purpose of the Temporal Attention layer is to capture the temporal dependencies of time series data. Its input is the time series data with missing values, mask vector, and time decay defined in the third section. The Relevance Attention layer captures the correlations between different features. Its input is the output of the Temporal Attention layer. Finally, the time information obtained from the Temporal Attention layer and the Feature Correlation information obtained from the Relevance Attention layer are inputted into the Feature Aggregation layer for aggregation to obtain the final output. The complete dataset obtained in this manner retains true values, replacing missing values with generated values, according to the calculation formula shown below.

$$x_{\text{imputed}} = \mathrm{M} \odot x + (1 - \mathrm{M}) \odot G(x) \tag{4}$$

Here, $x_{imputed}$ is the complete data set interpolated from the input data set x and the mask matrix M, which represents the distribution of missing values in the input data set. G(x) is the output of the generator.

The loss function of the generator includes two parts: adversarial loss and reconstruction loss. The adversarial loss is similar to that of a standard GAN.

The reconstruction loss is used to enhance the consistency between the observed time series and the reconstructed time series. The loss function of the generator is shown below:

$$\text{Loss}_G = \text{Loss}_g + \text{Loss}_r \tag{5}$$

$$\text{Loss}_g = \|x \odot M - x_{\text{imputed}} \odot M\|_2^2 \tag{6}$$

$$\text{Loss}_r = \log\left(1 - D\left(x_{\text{imputed}} \odot (1 - M)\right)\right) \tag{7}$$

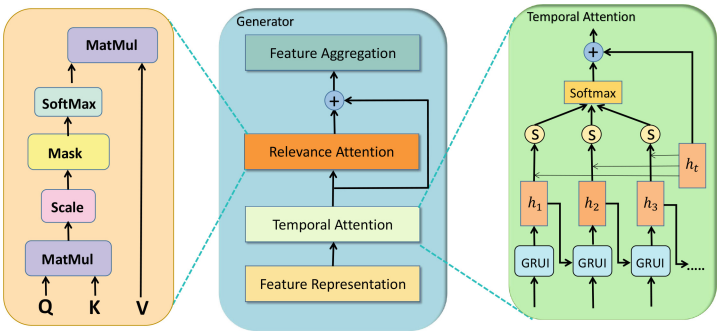


Fig. 2. The structure of DANGAN's Generator.

Improved GRU. A deeper structure is beneficial for a recurrent neural network to better model the time series structure, in order to capture more complex relationships in the time series [21]. Based on this, we designed a feature mapping module to represent the input data in terms of features and map them into potential representations. This can improve the learning ability of RNN without increasing complexity in aggregating at multiple time steps.

$$\boldsymbol{x}_t = \tanh\left(\boldsymbol{W}_m x + \boldsymbol{b}_m\right) \tag{8}$$

where the parameters W_m and b_m are the parameters to be learned, x represents the input data, tanh is a nonlinear activation function.

In our proposed model, we choose a gated recurrent neural network (GRU) to handle the time series input of the generation, which is a network structure adapted from the classical RNN and controls the information transfer in the neural network by adding a gating mechanism (nonlinear activation function, in this paper we use sigmoid activation function). GRU requires fewer parameters to train and converge faster. In a standard GRU model, the input to each GRU unit is the hidden state h_{t-1} of the previous unit's output and the current input x_t. Each GRU has an internal update gate and a reset gate. The data flow inside the GRU can be expressed as follows.

$$\mathbf{h}_t = (1 - \mu_t) \odot \mathbf{h}_{t-1} + \mu_t \odot \tilde{\mathbf{h}}_t \tag{9}$$

$$\tilde{\mathbf{h}}_t = \tanh\left(W_h \mathbf{x}_t + U_h\left(\mathbf{r}_t \odot \mathbf{h}_{t-1}\right) + \mathbf{b}_h\right) \tag{10}$$

$$\mu_t = \sigma\left(W_\mu \mathbf{x}_t + U_z \mathbf{h}_{t-1} + \mathbf{b}_\mu\right) \tag{11}$$

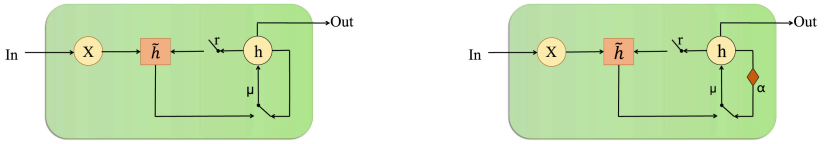
$$\mathbf{r}_t = \sigma\left(W_r \mathbf{x}_t + U_r \mathbf{h}_{t-1} + \mathbf{b}_r\right) \tag{12}$$

where μ_t and $\mathbf{r}_t$ are the update and reset gates of the GRU, respectively. tanh, σ, and $\odot$ denote the tanh activation function, the sigmod function, and the element multiplication.

We integrate the previously introduced Time Retention matrix and time decay factor α into the GRU unit. We multiply the time decay factor alpha with the GRU hidden state $\boldsymbol{h}_{t-1}$ to obtain the new hidden state $\boldsymbol{h}'_{t-1}$ (Fig. 3).

$$\boldsymbol{h}'_{t-1} = \boldsymbol{\alpha}_t \odot \boldsymbol{h}_{t-1} \tag{13}$$

$$\mathbf{h}_t = (1 - \mu_t) \odot \boldsymbol{h}'_{t-1} + \mu_t \odot \tilde{\mathbf{h}}_t \tag{14}$$



(a) An example of a typical GRU cell.

(b) Internal details of our improved GRU cell.

Fig. 3. Standard GRU Cell vs. Our Improved GRU Cell.

Temporal Attention. In order to solve the limitation of memory and excessive attention to adjacent time steps in the recurrent neural network when facing long time sequences, and to enhance the ability of the recurrent neural network to capture important information and long-term dependencies within the time series, we have designed a time recurrent attention mechanism. Our time recurrent attention mechanism weights the hidden states of each time step, and the weighted processing of attention will make the hidden states extracted from each time series contain comprehensive temporal information.

Given a set of hidden states $\mathrm{H} = \{\mathrm{h}_1,\ \mathrm{h}_2,\ \mathrm{h}_3 \ldots, \mathrm{h}_t\}$, calculating the importance score θ of each hidden state and then calculate the weighted sum of the hidden states to obtain the Context Vector v_t. In this way, we can effectively alleviate the disadvantage of GRU's tendency to forget the first few steps in long sequences [1]. The specific calculation formula is shown in Eqs. 15 and 16.

$$\theta_i = \frac{\exp\left(\mathrm{func}\left(\mathbf{h}_t, \mathbf{h}_i\right)\right)}{\sum_{j=1}^{t} \exp\left(\mathrm{func}\left(\mathbf{h}_t, \mathbf{h}_j\right)\right)} \tag{15}$$

$$\boldsymbol{v}_t = \sum_{i=1}^{t} \boldsymbol{\theta}_i \boldsymbol{h}_i \tag{16}$$

where func is the function that calculates the attention score between the current state and the historical hidden states. In this article, we use the hidden state at time t as the query vector and use dot product as the calculation function.

Relevance Attention. In multivariate time series data, there are different variables. By analyzing the correlations between different variables, the ultimate imputation accuracy can be improved. Therefore, in order to utilize the potential correlations between different variables, we designed a correlation attention mechanism based on the self-attention mechanism to capture the potential correlations between different variables. The self-attention mechanism is a mechanism used to calculate the representation of the sequence data. It automatically assigns different weights to each element in the sequence to better capture the relationships between different elements. The self-attention mechanism includes three parts: query, key and value, and calculates the attention representation vector of the sequence through similarity calculation, softmax function and weighted sum. The calculation formula of the self-attention mechanism is shown below.

$$\text{SelfAttention}(Q, K, V) = \text{Softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \tag{17}$$

In order to enhance the model's interpolation ability, we refer to the self-attention models in DISAN [20], SAITS [4], and XLNet [27]. We apply the diagonal mask matrix to the self-attention mechanism, and set the diagonal items in the attention map to $-\infty$. Therefore, the diagonal attention weight approaches 0 after the softmax function. We use vector $\mathbf{Z} \in \mathbb{R}^{d \times l}$ as inputs, and vector Z is stacked by Context Vector v_t. D is the number of variables in the multivariate time series datasert, and l is the length of v_t. The specific formula for relevance attention is as follows:

$$[\text{Mask}(x)]_{(i,j)} = \begin{cases} -\infty & i = j \\ x_{(i,j)} & i \neq j \end{cases} \tag{18}$$

$$\mathbf{Q} = \text{ZW}_\text{q} \quad \text{K} = \text{ZW}_\text{k} \quad \text{V} = \text{ZW}_\text{v} \tag{19}$$

$$T = \text{Softmax}\left(\text{Mask}\left(\frac{QK^\top}{\sqrt{d_k}}\right)\right) \tag{20}$$

$$\text{MaskSelfAttention}\ (Q, K, V) = TV \tag{21}$$

Q, K, and V respectively refer to the query vector, key vector, and value vector. W_q, W_k, and W_v are trainable parameters. In the calculation, we first use Q and K to calculate the similarity and then use the softmax function to process and obtain similarity scores, reflecting the correlation between different variables. By using a diagonal mask, the input values at time step t cannot 'see' themselves and are not allowed to calculate their own weight in this estimation. Therefore, their estimates only depend on the input values at other time steps. With this setting, we can better capture the relevance between different features and improve the model's interpolation ability using relevance attention.

Feature Aggregation Layer. We refer to the idea of residual connection, where we aggregate Z with temporal information, and $\hat{\mathbf{Z}}$ with correlation information, and finally obtain the final interpolation result after a linear layer.

$$\hat{x} = \text{Linear}(W(\mathbf{Z} + \hat{\mathbf{Z}}) + \text{b}) \tag{22}$$

where W, b are learnable parameters, $\mathbf{Z} \in \mathbb{R}^{d \times l}$ is a Context Vector with temporal information, and $\hat{\mathbf{Z}}$ is a correlation matrix with correlation information. The linear layer produces the final interpolated values.

4.3 Discriminator Network

Following the standard GAN model, we use a discriminator to compete with the generator, which helps the generator to generate more realistic data. Unlike general generative adversarial networks, our discriminator outputs a matrix in which each value indicates the truthfulness of the generated value. To help the discriminator can better distinguish the true values from the generated values, we introduce a temporal cueing matrix inspired by GAIN [28], which contains a portion of missing information. The temporal cueing matrix C is defined as:

$$\mathbf{C} = \mathbf{Y} \odot \mathbf{M} + 0.5(1 - \mathbf{K}) \tag{23}$$

$$\mathbf{Y} = (\mathbf{y}_1, \cdots, \mathbf{y}_i, \cdots, \mathbf{y}_n) \in \{0, 1\}^{d \times n} \tag{24}$$

where each element in y is randomly set to 0 or 1. The discriminator connects the generated time series data and the temporal cueing matrix as input, and the network structure of the discriminator consists of a GRU layer and a linear layer.

The loss function of the discriminator is shown below:

$$\text{Loss}_D = -\left(\log\left(D\left(x_{\text{impute}} \odot M\right)\right) + \log\left(1 - D\left(x_{\text{impute}} \odot (1 - M)\right)\right)\right) \tag{25}$$

where Loss_D refers to the classification loss of the discriminator, we want the discriminator to be able to distinguish the generated values from the true values as much as possible.

5 Experiment

5.1 Datasets

To validate the performance of our proposed model, we conducted experiments on two real datasets: PM2.5 Dataset, Health-care Dataset.

PM2.5 Dataset: This dataset is a public meteorological dataset consisting of air pollutant measurements from meteorological monitoring stations in Chinese cities. The data are collected from 2014-05-01 to 2017-02-28. To evaluate the interpolation performance, we first divide the dataset into a training set and a test set in the ratio of 80% and 20%, and then we randomly remove data from the dataset and use them as missing values for training and testing.

Health-care Dataset: This dataset is from PhysioNet Challenge 2012 [5], and it contains 4000 multivariate clinical time series data, each sample is recorded within the first 48 h after ICU admission. Each time series contains 37 time series variables such as body temperature, heart rate, blood pressure, etc. During training, we will randomly remove data points in the dataset as missing values and then use zeros to fill these missing values.

5.2 Baseline

This section describes the methods and models commonly used in time series interpolation and applies them to the previously mentioned dataset, and finally compares them with the model proposed in this paper.

1) Zero, a simple method of data interpolation, which fills all missing values to zero;
2) Average, where the missing values are replaced by the average of the corresponding features;
3) GRUD [3], a recurrent neural network based data interpolation model which estimates each missing value by the weighted combination of its last observation and the global average and the recurrent component;
4) GRUIGAN [15], an interpolation network combining gated recurrent network and generative adversarial network;
5) BRITS [2], a time series imputation method based on bi-directional RNN;
6) E2GAN [16], an end-to-end time series interpolation model based on generative adversarial networks;
7) SSGAN [17], a semi-supervised generative adversarial network based on bidirectional RNN.

Among the above methods, mean is a simple interpolation method. KNN is a commonly used algorithm for interpolating machine learning data. GRUD and BRITS are both bi-directional RNN-based methods. GRUIGAN, E2GAN, SSGAN are all generative adversarial network based models, we choose these as comparison methods to show the advantages of our model.

5.3 Experimental Setup

On the above dataset, we select 80% of the dataset as the training set and 20% of the dataset as the test set. For all tasks, we normalize the values to ensure stable training. In all the deep learning baseline models, the learning rate is set to 0.001. The number of hidden units in the recurrent network is 100, and the training epoch is set to 30. The dimensionality of random noise in GRUIGAN and the dimensionality of feature vectors in E2GAN are both 64. For SSGAN, we set the cue rate to 0.9 and the label rate to 100%. For the DAGAN model in thiTs paper, the discriminator cue rate is set to 0.9 and the hidden layer cells are set to 100. We apply an early stopping strategy in the model training. We also evaluate the performance of all models with interpolation at different missing

rates. The missing rate is the ratio of missing values to the total number of data. It reflects the severity of missingness in the dataset. We randomly remove 10%–70% of the test data points to simulate different degrees of missingness. We use the root mean square error (RMSE) to evaluate the experimental results. A smaller value of RMSE means that the generated value is closer to the true value, and the following is the mathematical definition of the evaluation metric.

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(\text{ target-estimate })^2} \tag{26}$$

where target and estimate are the true and generated values, respectively, and m is the number of samples.

5.4 Experimental Results and Analysis

Table 1 displays the experimental results of the model with PM2.5 and Healthcare Datasets. Bolded results indicate the best-performing. The first set of experiments compares the performance of various methods at different missing data rates. DAGAN performs well on all datasets, demonstrating good generalization capability. DAGAN outperforms the best available baseline model on average by 13.8%, 12.2%, 8.07%, and 3.9%, respectively, at 10%, 30%, 50%, and 70% missing rates. Its use of temporal attention and relevance attention captures temporal correlation and potential correlation between features to improve interpolation accuracy by leveraging the available information from the time series. Overall, deep learning-based approaches exhibit better interpolation performance compared to statistics-based approaches. Table 1 reveals that the interpolation accuracy of all models declines as missing data increases gradually. This can be attributed to the reduced availability of information for the models to interpolate. Nonetheless, our models still outperform the baseline model.

5.5 Ablation Experiment

To ensure the validity of our model, we conducted ablation experiments, each repeated ten times, and recorded the average root mean square error (RMSE) at a 10% missing rate for the test set. The ablation experiments evaluated the model's performance by removing the temporal attention mechanism, the relevance attention mechanism, and the temporal cueing mechanism, respectively. Our experimental results reveal that the removal of the temporal attention mechanism, the relevance attention mechanism, and the temporal cueing mechanism led to a reduction in performance, indicating that the optimal use of temporal and correlation information between distinct features in the dataset is crucial for achieving high interpolation accuracy. Additionally, the results indicate that our adversarial training benefits from the incorporation of a temporal cueing matrix (Table 2).

Table 1. Performance comparison of time series imputation methods under different missing rates.

Dataset	Missing	Zero	Average	GRUD	GRUIGAN	E2GAN	BRITS	SSGAN	DAGAN
Health-care Dataset	10%	0.793	0.786	0.695	0.674	0.661	0.593	0.586	**0.519**
	30%	0.832	0.847	0.761	0.758	0.725	0.657	0.651	**0.639**
	50%	0.904	0.895	0.793	0.787	0.763	0.757	0.746	**0.722**
	70%	0.934	0.929	0.826	0.804	0.801	0.793	0.776	**0.749**
PM2.5 Dataset	10%	0.779	0.758	0.695	0.685	0.653	0.528	0.429	**0.367**
	30%	0.804	0.799	0.763	0.748	0.668	0.563	0.489	**0.394**
	50%	0.890	0.871	0.793	0.779	0.731	0.574	0.487	**0.423**
	70%	0.921	0.903	0.812	0.786	0.762	0.638	0.601	**0.575**

Table 2. The results of ablation experiments (RMSE).

	Health-care Dataset	PM2.5 Dataset
DAGAN	**0.519**	**0.367**
w/o cue matrix	0.579	0.412
w/o temporal attention	0.683	0.503
w/o multi-head self-attention	0.581	0.393

6 Conclusion

In this paper, we propose a new multivariate time series interpolation model called DAGAN. For time series data, DAGAN consists of two parts: a generator and a discriminator. In the generator, we use a gated recurrent neural network to learn the temporal information in the multivariate time series data and use an attention mechanism to weight the summation of the hidden states in the recurrent neural network, which enhances the ability of the model to learn the long-term dependence of the time series while ensuring that the model can focus more on the important information and compensate for the disadvantages of memory limitation and excessive focus on adjacent time steps of the recurrent neural network, thus improving the interpolation quality. In addition, this study also utilizes a masked self-attentiveness mechanism to correlate different variables in the multivariate time series so that all time steps are involved in each layer through self-attentiveness, thus maximizing the accuracy of multivariate time series interpolation. The generator inputs incomplete data with missing values and outputs the complete interpolation, and the discriminator heaps the generated values and the true values to distinguish them. Experimental results demonstrate that, when compared with other baseline models, our model's imputation performance is better.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
2. Cao, W., Wang, D., Li, J., Zhou, H., Li, L., Li, Y.: Brits: bidirectional recurrent imputation for time series. In: Advances in Neural Information Processing Systems, vol. 31 (2018)
3. Che, Z., Purushotham, S., Cho, K., Sontag, D., Liu, Y.: Recurrent neural networks for multivariate time series with missing values. Sci. Rep. **8**(1), 6085 (2018)
4. Du, W., Côté, D., Liu, Y.: Saits: self-attention-based imputation for time series. Expert Syst. Appl. **219**, 119619 (2023)
5. Feng, F., Chen, H., He, X., Ding, J., Sun, M., Chua, T.S.: Enhancing stock movement prediction with adversarial training. arXiv preprint arXiv:1810.09936 (2018)
6. Fortuin, V., Baranchuk, D., Rätsch, G., Mandt, S.: Gp-vae: deep probabilistic time series imputation. In: International Conference on Artificial Intelligence and Statistics, pp. 1651–1661. PMLR (2020)
7. Fung, D.S.: Methods for the estimation of missing values in time series (2006)
8. Gupta, M., Phan, T.L.T., Bunnell, H.T., Beheshti, R.: Concurrent imputation and prediction on EHR data using bi-directional GANs: Bi-GANs for EHR imputation and prediction. In: Proceedings of the 12th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics, pp. 1–9 (2021)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
10. Hudak, A.T., Crookston, N.L., Evans, J.S., Hall, D.E., Falkowski, M.J.: Nearest neighbor imputation of species-level, plot-scale forest structure attributes from lidar data. Remote Sens. Environ. **112**(5), 2232–2245 (2008)
11. Kaiser, J.: Dealing with missing values in data. J. Syst. Integr. (1804–2724) **5**(1) (2014)
12. Lan, Q., Xu, X., Ma, H., Li, G.: Multivariable data imputation for the analysis of incomplete credit data. Expert Syst. Appl. **141**, 112926 (2020)
13. LIU, S., Li, X., Cong, G., Chen, Y., Jiang, Y.: Multivariate time-series imputation with disentangled temporal representations. In: The Eleventh International Conference on Learning Representations (2023)
14. Liu, X., Wang, M.: Gap filling of missing data for VIIRS global ocean color products using the DINEOF method. IEEE Trans. Geosci. Remote Sens. **56**(8), 4464–4476 (2018)
15. Luo, Y., Cai, X., Zhang, Y., Xu, J., et al.: Multivariate time series imputation with generative adversarial networks. In: Advances in Neural Information Processing Systems, vol. 31 (2018)
16. Luo, Y., Zhang, Y., Cai, X., Yuan, X.: E2gan: end-to-end generative adversarial network for multivariate time series imputation. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence, pp. 3094–3100. AAAI Press (2019)
17. Miao, X., Wu, Y., Wang, J., Gao, Y., Mao, X., Yin, J.: Generative semi-supervised learning for multivariate time series imputation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 8983–8991 (2021)
18. Ni, Q., Cao, X.: MBGAN: an improved generative adversarial network with multi-head self-attention and bidirectional RNN for time series imputation. Eng. Appl. Artif. Intell. **115**, 105232 (2022)

19. Qin, R., Wang, Y.: ImputeGAN: generative adversarial network for multivariate time series imputation. Entropy **25**(1), 137 (2023)
20. Shen, T., Zhou, T., Long, G., Jiang, J., Pan, S., Zhang, C.: Disan: Directional self-attention network for rnn/cnn-free language understanding. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
21. Silva, I., Moody, G., Scott, D.J., Celi, L.A., Mark, R.G.: Predicting in-hospital mortality of ICU patients: the physionet/computing in cardiology challenge 2012. In: 2012 Computing in Cardiology, pp. 245–248. IEEE (2012)
22. Suo, Q., Yao, L., Xun, G., Sun, J., Zhang, A.: Recurrent imputation for multivariate time series with missing values. In: 2019 IEEE International Conference on Healthcare Informatics (ICHI), pp. 1–3. IEEE (2019)
23. Suo, Q., Zhong, W., Xun, G., Sun, J., Chen, C., Zhang, A.: Glima: global and local time series imputation with multi-directional attention learning. In: 2020 IEEE International Conference on Big Data (Big Data), pp. 798–807. IEEE (2020)
24. Tang, F., Ishwaran, H.: Random forest missing data algorithms. Stat. Anal. Data Min.: ASA Data Sci. J. **10**(6), 363–377 (2017)
25. Wang, R., Zhang, Z., Wang, Q., Sun, J.: TLGRU: time and location gated recurrent unit for multivariate time series imputation. EURASIP J. Adv. Signal Process. **2022**(1), 74 (2022)
26. Woodall, P.: The data repurposing challenge: new pressures from data analytics. J. Data Inf. Qual. (JDIQ) **8**(3–4), 1–4 (2017)
27. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: Xlnet: generalized autoregressive pretraining for language understanding. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
28. Yoon, J., Jordon, J., Schaar, M.: Gain: missing data imputation using generative adversarial nets. In: International Conference on Machine Learning, pp. 5689–5698. PMLR (2018)