Enhancing Imputation Performance in Univariate Time Series Using Gated Recurrent Unit

Tan Thai Nhat
Department of Artificial Intelligence
FPT University
Danang, Vietnam
0009-0005-6205-0366

Trinh Le Nhat
Department of Artificial Intelligence
FPT University
Danang, Vietnam
0009-0007-4405-5280

Thi-Thu-Hong Phan*
Department of Artificial Intelligence
FPT University
Danang, Vietnam
0000-0001-6880-3721
*Corresponding author: hongptt11@fe.edu.vn

***Abstract*—Missing data is a pervasive problem in time series analysis, often leading to biased predictions and compromised model performance. This study introduces the Gated Recurrent Unit (GRU) as a robust solution for estimating missing values in time series data. Experimental results demonstrate GRU's superior performance compared to traditional machine learning methods like Random Forest, Support Vector Regression, Extra Trees, and K-Nearest Neighbors. GRU's exceptional ability to capture complex, dynamic relationships in time series data makes it an ideal choice for accurate imputation, ensuring the reliability of predictive models for time series data.**

***Index Terms*—Gated Recurrent Unit (GRU), Machine learning methods, Missing data, Univariate time series, Imputation.**

I. Introduction

Time series data, prevalent in various domains, often suffer from missing values due to data collection issues. Traditional imputation methods like interpolation [9], Last Observation Carried Forward (LOCF) [13], and mean/median/mode imputation [7], while efficient, are popular but can lose temporal information and ignore variable relationships, particularly with long gaps of missing data.

To improve the limitations of traditional methods, machine learning (ML) techniques such as Random Forest (RF) [19], Support Vector Machine (SVM) [11], K-Nearest Neighbors (KNN) [17], and Extra Tree (ET) [18] enhance imputation accuracy by capturing nonlinear relationships. However, they may underutilize spatial relationships, be computationally intensive, and complex to implement. While not traditionally classified under machine learning, Dynamic Time Warping-based Imputation (DTWBI) [15] is recognized as an effective method that yields promising results for imputing missing data in univariate time series, albeit at the cost of significant computational time.

Deep learning algorithms present a powerful alternative to overcome the limitations inherent in traditional and machine learning methods. Among these, Recurrent Neural Networks (RNNs) [10], [21] have shown particular promise. RNNs, such as Long Short-Term Memory (LSTM) [8], [20] and Gated Recurrent Unit (GRU) [5], [8], [16], are well-suited for handling time-series data and capturing long-term dependencies. While LSTM excels at capturing long-term dependencies, its computational complexity can be a drawback. Conversely, GRU offers a more efficient alternative by using fewer gates but may be less flexible in certain scenarios. Despite the promising potential of the GRU model for time series data, it has not been widely applied to the imputation task. Therefore, in this study, we aim to explore the application of GRU for completing missing data in univariate time series to assess its effectiveness in this context.

The paper is organized as follows: Section 2 describes the methodology, Section 3 presents the experimental setup and results, and Section 4 presents conclusions with future work suggestions.

II. Methodology

While still grounded in the framework of the previous study [14], we have introduced a significant modification in the prediction stage by employing GRU instead of ML methods in this work. This enables us to fully exploit the temporal information inherent in the data. A concise description of the steps is provided below:

Data Division: Aiming to fully exploit the information from the data, we have adopted a comprehensive approach, including considering the data before and after the gaps. Therefore, we have established different rules for different cases: For missing data segments at the beginning or end of the time series, we focus solely on forecasting based on the remaining data. In other cases, we consider both the data before and after the gap for prediction.

Data Transformation: In this step, we transform the univariate data into a multivariate format with T dimensions, where T represents the length of the missing data. This transformation enables us to effectively utilize machine learning/deep learning algorithms to estimate the missing values.

Training Model: Following the data transformation, we employ the Gated Recurrent Unit (GRU) model, which is particularly well-suited for handling time series data. To evaluate its effectiveness, we compare its performance against other machine learning models, including Random Forest, Support Vector Machine, K-Nearest Neighbors, and Extra Trees

Forecasting Missing Values: Missing data is estimated using a one-step-ahead forecasting approach, where each value is predicted based on previous forecasts. This process is applied iteratively in both forward and backward directions until all gaps are filled.

Filling Missing Values: The final imputed values are computed by averaging the outcomes from both forward and backward imputations.

A. Gated Recurrent Unit (GRU)

The GRU, introduced in [2], is a simplified variant of the LSTM network with fewer parameters, which accelerates training while maintaining effective performance.

The GRU incorporates gating mechanisms to control the information flow and selectively update the hidden state at each time step. The two gates—reset and update—manage how much of the previous hidden state is forgotten and how much of the new input is used to update the hidden state. The updated hidden state determines the GRU's output, as illustrated in Figure 1.

Given an input vector $\mathbf{X}_t$ at time step t, and the hidden state $\mathbf{H}_{t-1}$ from the previous time step $t-1$, the GRU updates the hidden state as follows:

1) **Reset Gate**:

$$\mathbf{R}_t = \sigma(\mathbf{W}_{xr}\mathbf{X}_t + \mathbf{W}_{hr}\mathbf{H}_{t-1} + \mathbf{b}_r)$$

2) **Update Gate**:

$$\mathbf{Z}_t = \sigma(\mathbf{W}_{xz}\mathbf{X}_t + \mathbf{W}_{hz}\mathbf{H}_{t-1} + \mathbf{b}_z)$$

3) **Candidate Hidden State**:

$$\tilde{\mathbf{H}}_t = \tanh(\mathbf{W}_{xh}\mathbf{X}_t + \mathbf{R}_t \odot (\mathbf{W}_{hh}\mathbf{H}_{t-1}) + \mathbf{b}_h)$$

4) **Hidden State Update**:

$$\mathbf{H}_t = (1 - \mathbf{Z}_t) \odot \mathbf{H}_{t-1} + \mathbf{Z}_t \odot \tilde{\mathbf{H}}_t$$

In these equations: - $\mathbf{W}_{xr}, \mathbf{W}_{xz}, \mathbf{W}_{xh}$ are weight matrices for the input vector $\mathbf{X}_t$. - $\mathbf{W}_{hr}, \mathbf{W}_{hz}, \mathbf{W}_{hh}$ are weight matrices for the hidden state $\mathbf{H}_{t-1}$. - $\mathbf{b}_r, \mathbf{b}_z, \mathbf{b}_h$ are bias terms. - σ denotes the sigmoid activation function. - $\tanh$ is the hyperbolic tangent activation function. - $\odot$ represents the element-wise multiplication (Hadamard product).

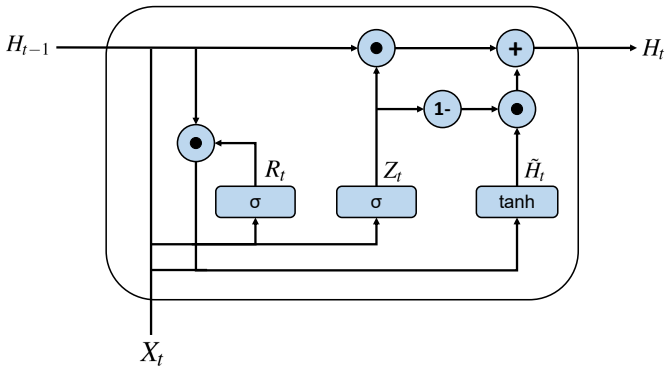

Fig. 1. GRU architechture

B. Machine Learning Models

Random Forest (RF): is a robust ensemble learning method utilized for both classification and regression tasks [1]. The method constructs multiple decision trees during training and for classification problems, the output is determined by the class that appears most frequently across all predictions. In contrast, for regression problems, the final result is computed as the mean of the predicted values. To reduce correlation among trees, Random Forest employs bagging, which involves generating bootstrap samples from the original data. Additionally, during node splitting in tree construction, only a random subset of features is considered, rather than all available features. This randomness helps mitigate overfitting and enhances model generalization.

Support Vector Machine (SVM): is a powerful algorithm frequently used for both classification and regression tasks [3]. SVMs are particularly effective in handling nonlinear relationships and high-dimensional data. For forecasting and imputation, which are regression problems, Support Vector Regression (SVR) is often employed. SVR seeks to identify an optimal hyperplane in a high-dimensional space that best fits the training data while minimizing prediction errors. This approach enables SVR to capture complex patterns and provide accurate forecasts or imputed values.

K-Nearest Neighbors (KNN): is a non-parametric, instance-based learning algorithm known for its simplicity and effectiveness in various contexts [4]. Unlike many machine learning models, KNN does not involve an explicit training phase. Instead, it performs computations at the time of prediction. The algorithm identifies the K nearest data points to a given query point using a specified distance metric. For classification tasks, the predicted class is determined by the majority class among these K neighbors, while in regression, the predicted value is the average of the target values of the K nearest neighbors. This adaptability makes KNN suitable for a range of prediction problems.

Extra Trees (ET): is an ensemble learning algorithm designed for both regression and classification tasks [6]. Although it is similar to Random Forests, Extra Trees introduces additional randomness to enhance efficiency and model diversity. Unlike Random Forests, which select a subset of features at each node, Extra Trees consider all features and choose split points completely at random. This extreme level of randomization helps reduce overfitting and often leads to performance that is comparable to, or even surpasses, that of Random Forests with fewer trees. Consequently, Extra Trees are computationally efficient and robust, making them a valuable tool in machine learning applications due to their speed, accuracy, and adaptability.

III. Experiments and Results

In this section, we begin by detailing the datasets used in our experiments. Next, we present the performance evaluation metrics employed to assess our models. Finally, we discuss the experimental results, highlighting key findings and observations.

A. Data description

In this study, we utilize water level data from the Hanoi hydrology station along the Red River, which is the primary river in the Northern Delta of Vietnam. The data, comprising 29,224 samples, were recorded every three hours from January 1, 2008, to December 31, 2017. Figure 2 illustrates the water level data collected from the Hanoi hydrology station.

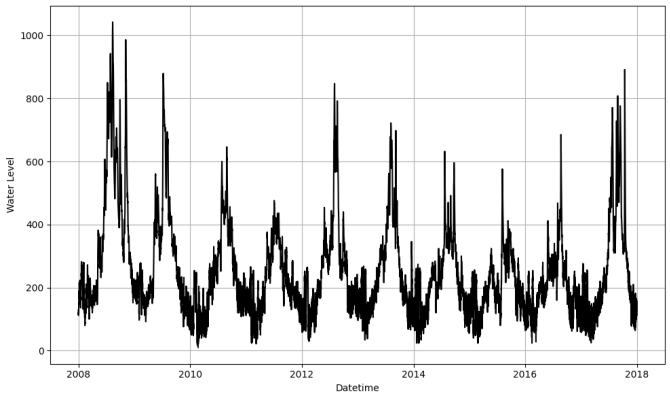

Fig. 2. Water level at Hanoi hydrology stations of the Red River

B. Evaluation metrics

To evaluate the performance of imputation algorithms, we use five criteria:

- **Sim (Similarity)**: Measures the similarity between imputed values (Y) and true values (X). The Sim value ranges from 0 to 1, with 1 indicating perfect similarity. It is defined as:

$$\text{Sim}(Y, X) = \frac{1}{T} \sum_{i=1}^{T} \frac{1}{1 + \frac{|y_i - x_i|}{\max(X) - \min(X)}} \tag{1}$$

- **MAE (Mean Absolute Error)**: Represents the average absolute difference between predicted ($\hat{y}_i$) and actual values (y_i). A lower MAE indicates better accuracy:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{2}$$

- **RMSE (Root Mean Squared Error)**: The square root of the average of squared differences between predicted and actual values. A lower RMSE signifies higher accuracy:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \tag{3}$$

- **FSD (Fraction of Standard Deviation)**: Compares the variability of imputed values to that of actual values. It is calculated as:

$$\text{FSD} = 2 \times \left| \frac{SD(Y) - SD(X)}{SD(Y) + SD(X)} \right| \tag{4}$$

- **NSE (Nash-Sutcliffe Efficiency)**: Evaluates the performance of imputation models, ranging from $-\infty$ to 1, with values closer to 1 indicating better performance [12]:

$$\text{NSE} = 1 - \frac{\sum_{i=1}^{T} (x_i - y_i)^2}{\sum_{i=1}^{T} (x_i - \bar{x})^2} \tag{5}$$

C. Experiment results

This section presents the outcomes of our proposed approach in comparison with various imputation methods. To evaluate performance in the absence of actual missing values, we artificially introduced gaps into the time series. We have tested five different missing data rates in the time series dataset, creating gaps at ten random locations for each rate. The performance metrics are then averaged across these gaps, with each gap size (12, 24, 48, 72, and 120 hours) imputed ten times. In this study, we have optimized the GRU model with 500 epochs, early stopping at 100 epochs, a batch size of 256, and a 20% validation split for performance monitoring

Table I presents a comparative analysis of various machine learning methods used to estimate missing data in the Hanoi time series dataset. The evaluation of these methods is based on five metrics such as Sim, MAE, RMSE, FSD, and NSE. The models are evaluated on different forecast horizons (12h, 24h, 48h, 72h, 120h).

Overall, the GRU model shows the best performance across most metrics and forecast horizons. This suggests that GRU is better at capturing temporal patterns and making accurate predictions. The RF model also performs well, especially for shorter forecast horizons. RF is good at handling non-linear data and is commonly used in many prediction problems. SVM and KNN models generally perform worse compared to GRU and RF, especially for longer forecast horizons. This might be due to SVM and KNN not being as suitable for highly correlated time series data.

For 12-hour gaps: GRU outperforms other methods with the highest similarity (0.81) and the lowest MAE (16.82) and RMSE (19.41). It also has a notable FSD of 0.77 and NSE of 0.04. Other methods like RF and SVM show lower performance in comparison, particularly in MAE and RMSE.

For 24-hour gaps: GRU continues to lead with the highest similarity (0.82) and the lowest MAE (19.98) and RMSE (23.28). It also shows the best FSD (0.79) and a slight negative NSE (-0.07), indicating better performance than RF, SVM, ET, and KNN.

For 48-hour gaps: GRU maintains its strong performance with the highest similarity (0.83) and the lowest MAE (23.96) and RMSE (29.93). It has a lower FSD of 0.68 and NSE of -0.6, showing its effectiveness over other methods like RF and SVM, which have higher MAE and RMSE.

For 72-hour gaps: GRU again shows the best similarity (0.83) and the lowest RMSE (29.72), alongside a competitive MAE (24.56). Its FSD (0.67) and NSE (-0.63) are also favorable compared to methods like RF, SVM (RBF), and KNN, which exhibit less accuracy and higher errors.

For 120-hour gaps: GRU continues to demonstrate superior performance with the highest similarity (0.83) and the lowest RMSE (42.79). Its MAE (35.61) is also among the lowest, with an FSD of 0.83 and NSE of -0.44. In comparison, RF performs similarly in similarity (0.83) but shows higher MAE and RMSE, while methods like SVM (Linear) have notably worse performance.

TABLE I
PERFORMANCE OF ML METHODS FOR ESTIMATING MISSING DATA IN HANOI TIME SERIES

Methods	Sim	MAE	RMSE	FSD	NSE
12h					
GRU	**0.81**	**16.82**	**19.41**	**0.77**	**0.04**
RF	0.75	19.88	22.71	1.05	-1.17
SVM (RBF)	0.74	22.96	25.74	1.22	-2.61
SVM (Linear)	0.75	21.37	24.13	1.19	-2.02
ET	0.76	20.5	23.97	0.96	-1.02
KNN	0.74	21.5	24.92	0.89	-2.96
24h					
GRU	**0.82**	**19.98**	**23.28**	**0.79**	**-0.07**
RF	0.77	31.8	36.7	1.13	-2.37
SVM (RBF)	0.81	31.25	36.17	1.1	-0.08
SVM (Linear)	0.8	31.8	36.39	1.09	-0.23
ET	0.78	31.95	37.27	1.02	-1.13
KNN	0.8	29.88	34.63	0.91	-0.22
48h					
GRU	**0.83**	**23.96**	**29.93**	0.68	-0.6
RF	0.81	26.63	31.8	1.09	-0.46
SVM (RBF)	0.81	27.03	31.57	0.93	-0.55
SVM (Linear)	0.79	31.43	35.97	1.11	-1.6
ET	0.81	26.65	31.27	0.99	**-0.42**
KNN	0.8	30.89	35.88	**0.64**	-1.09
72h					
GRU	**0.83**	**24.56**	**29.72**	0.67	**-0.63**
RF	0.8	29.56	34.16	1.07	-0.97
SVM (RBF)	0.74	42.24	48.45	**0.57**	-3.17
SVM (Linear)	0.81	27.34	32.55	0.94	-0.74
ET	0.82	26.18	31.64	0.94	-0.76
KNN	0.72	49.38	56.98	0.56	-6.29
120h					
GRU	**0.83**	**35.61**	**42.79**	0.83	-0.44
RF	**0.83**	38.55	46.63	1.16	**-0.37**
SVM (RBF)	0.76	56.63	66.1	0.77	-4.49
SVM (Linear)	0.76	123.92	163.4	1.29	-178.12
ET	0.82	42.31	50.26	0.93	-0.63
KNN	0.8	42.62	51.99	**0.58**	-2.64

IV. CONCLUSION

This study explores the application of the advanced deep learning algorithm to the critical task of imputing missing data in time series. By transforming univariate data into a multivariate structure and leveraging both pre-gap and post-gap temporal patterns, we develop a robust framework for multi-step-ahead predictions based on one-step-ahead. The deep learning model, GRU, consistently outperforms machine learning methods, regardless of the length of the missing data. These findings highlight the effectiveness of GRU in addressing the challenges posed by missing data in time series analysis. Future research will focus on integrating multiple deep learning algorithms to capitalize on their respective strengths and further enhance imputation performance

REFERENCES

[1] Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)
[2] Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling (2014), https://arxiv.org/abs/1412.3555
[3] Cortes, C., Vapnik, V.: Support-vector networks. Machine learning **20**(3), 273–297 (1995)
[4] Cover, T., Hart, P.: The nearest neighbor decision rule. IEEE Transactions on Information Theory (1967)
[5] Dutta, A., Kumar, S., Basu, M.: A gated recurrent unit approach to bitcoin price prediction. Journal of risk and financial management **13**(2), 23 (2020)
[6] Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine learning **63**(1), 3–42 (2006)
[7] Jadhav, A., Pramod, D., Ramanathan, K.: Comparison of performance of data imputation methods for numeric dataset. Applied Artificial Intelligence **33**(10), 913–933 (2019)
[8] Lawi, A., Mesra, H., Amir, S.: Implementation of long short-term memory and gated recurrent units on grouped time-series data to predict stock prices accurately. Journal of Big Data **9**(1), 89 (2022)
[9] Lepot, M., Aubin, J.B., Clemens, F.H.: Interpolation in time series: An introductive overview of existing methods, their performance criteria and uncertainty assessment. Water **9**(10), 796 (2017)
[10] Madan, R., Mangipudi, P.S.: Predicting computer network traffic: a time series forecasting approach using dwt, arima and rnn. In: 2018 Eleventh International Conference on Contemporary Computing (IC3). pp. 1–5. IEEE (2018)
[11] Mallinson, H., Gammerman, A.: Imputation using support vector machines. University of London Egham, UK: Department of Computer Science Royal Holloway (2003)
[12] Nash, J.E., Sutcliffe, J.V.: River flow forecasting through conceptual models part i – a discussion of principles. Journal of Hydrology **10**(3), 282–290 (1970)
[13] Overall, J.E., Tonidandel, S., Starbuck, R.R.: Last-observation-carried-forward (locf) and tests for difference in mean rates of change in controlled repeated measurements designs with dropouts. Social Science Research **38**(2), 492–503 (2009)
[14] Phan, T.T.H.: Machine learning for univariate time series imputation. In: 2020 International Conference on Multimedia Analysis and Pattern Recognition (MAPR). pp. 1–6. IEEE (2020)
[15] Phan, T.T.H., Émilie Poisson Caillault, Lefebvre, A., Bigand, A.: Dynamic time warping-based imputation for univariate time series data. Pattern Recognition Letters **139**, 139–147 (November 2020). https://doi.org/10.1016/j.patrec.2017.08.019
[16] Shen, G., Tan, Q., Zhang, H., Zeng, P., Xu, J.: Deep learning with gated recurrent unit networks for financial sequence predictions. Procedia computer science **131**, 895–903 (2018)
[17] Sridevi, S., Rajaram, S., Parthiban, C., SibiArasan, S., Swadhikar, C.: Imputation for the analysis of missing values and prediction of time series data. In: 2011 International Conference on Recent Trends in Information Technology (ICRTIT). pp. 1158–1163 (2011)
[18] Suresh, M., Taib, R., Zhao, Y., Jin, W.: Sharpening the blade: missing data imputation using supervised machine learning. In: Australasian Joint Conference on Artificial Intelligence. pp. 215–227. Springer (2019)
[19] Yang, J.H., Cheng, C.H., Chan, C.P.: A time-series water level forecasting model based on imputation and variable selection method. Computational intelligence and neuroscience **2017**(1), 8734214 (2017)
[20] Yuan, H., Xu, G., Yao, Z., Jia, J., Zhang, Y.: Imputation of missing data in time series for air pollutants using long short-term memory recurrent neural networks. In: Proceedings of the 2018 ACM international joint conference and 2018 international symposium on pervasive and ubiquitous computing and wearable computers. pp. 1293–1300 (2018)
[21] Zhang, J., Man, K.F.: Time series prediction using rnn in multi-dimension embedding phase space. In: SMC'98 conference proceedings. 1998 IEEE international conference on systems, man, and cybernetics (cat. no. 98CH36218). vol. 2, pp. 1868–1873. IEEE (1998)