A lightweight 1DCNN for univariate time series imputation

Thi-Thu-Hong Phan[0000−0001−6880−3721]⋆, Tan Thai Nhat[0009−0005−6205−0366], Trinh Le Nhat[0009−0007−4405−5280]

Department of Artificial Intelligence, FPT University, Da Nang, Vietnam

Abstract. Imputation of missing data in time series is a critical challenge that significantly impacts the accuracy and reliability of predictive models across various fields. This study investigates the effectiveness of a self-defined lightweight 1-dimensional Convolutional Neural Network (CNN) model compared to traditional Machine Learning (ML) methods for addressing this issue. By transforming univariate time series data into a multivariate format, we exploit the inherent temporal relationships within the dataset to enhance the imputation of missing values. We evaluate the performance of the CNN model against several ML algorithms, including Random Forest, Support Vector Regression, Extra Trees, and K-Nearest Neighbors, using datasets from Vu Quang and Hanoi. Experimental results consistently demonstrate that the CNN model outperforms traditional ML methods across various gap sizes, achieving lower Mean Absolute Error (MAE) and Root Mean Square Error (RMSE), as well as higher Similarity, Nash-Sutcliffe Efficiency (NSE) with actual values. These findings underscore the superiority of CNN as a robust solution for time series data imputation, offering a more effective approach for handling missing data in predictive modeling tasks.

Keywords: 1DCNN · Missing values · Imputation · Univariate time series · Bidirectional

1 Introduction

Time series data is an integral part of our modern lives, capturing information continuously over fixed intervals such as hours, days, months, or years. Examples include financial data like stock prices, environmental data such as temperature, health indicators like heart rate, etc. Methods for collecting time series data vary widely, from manual recording to automated sensor devices and APIs that gather data from online sources.

However, data collection processes are prone to errors, resulting in missing data. These gaps can be caused by various factors such as transmission errors, device malfunctions, or simply gaps in the data recording process. Missing data poses a significant challenge in data processing, analysis, and evaluation. Such gaps can compromise the accuracy of analyses and assessments, reducing the

⋆ Corresponding author: hongptt11@fe.edu.vn

performance and reliability of forecasting models. This limits our ability to make accurate predictions about future trends.

Traditional imputation methods, such as interpolation [9], Last Observation Carried Forward (LOCF) [13], and mean/median/mode imputation [6], have been widely used to address missing values in time series data. ARIMA (AutoRegressive Integrated Moving Average) is another commonly employed statistical method, combining autoregressive (AR), differencing (I), and moving average (MA) components to model and forecast time series data. ARIMA assumes a linear relationship in the data and requires users to specify the model structure, making it suitable for simple, linear time series. According to Shaadan et al. (2019) [15], combining ARIMA with Kalman filtering achieved a notably low RMSE (9.43-9.45), outperforming some methods proposed in their study. However, while these approaches can be effective in certain scenarios, they often introduce biases, distort underlying patterns, and discard valuable information, leading to suboptimal imputation results.

To overcome the limitations of traditional methods, more sophisticated techniques, particularly those leveraging machine learning and deep learning, have emerged. These advanced approaches have demonstrated superior performance in handling missing data in time series, providing more accurate and reliable imputations. Popular machine learning methods such as Random Forest (RF) [18], Support Vector Machine (SVM) [10], K-Nearest Neighbors (KNN) [16], and Extra Tree (ET) [17] are now applied to time series imputation. These algorithms can handle complex nonlinear relationships in the data, offering more precise and dependable estimates compared to traditional methods. However, machine learning algorithms also have some drawbacks, including not fully exploiting spatial relationships, requiring significant computational resources, being more complex to implement and interpret, and, in some cases, taking more time to execute.

To address these limitations and achieve more accurate and efficient results, deep learning networks have emerged as a powerful solution. Among these, Convolutional Neural Networks (CNNs) have proven particularly effective for time series data. CNNs, originally designed for image processing, have been adapted to handle the sequential nature of time series data by applying convolutional layers that can capture spatial patterns and temporal dependencies within the data. This adaptation allows CNNs to efficiently learn and extract features from time series data, enhancing predictive performance and providing more robust insights compared to traditional methods [20, 19].

The success of CNNs in forecasting and imputation studies, as demonstrated in [7, 4], has proven their effectiveness in handling time series data. To address the prevalent issue of missing data in univariate time series, we have developed a lightweight and efficient 1DCNN model. This model not only inherits the advantages of CNNs but also excels in accurately estimating missing values.

The rest of the paper is organized as follows: In Section 2, we present the methodology employed in this study. Section 3 discusses the experimental settings and results. Finally, Section 4 provides concluding remarks and future perspectives.

2 Methodology

Building on our previous research [14], this study introduces a significantly enhanced approach to imputing missing values in univariate time series, aimed at achieving more accurate and robust results. To capture the complex temporal dynamics in time series data, we propose a lightweight one-dimensional convolutional neural network (1DCNN) model, which efficiently extracts relevant features and outperforms traditional machine learning methods.

We continue to utilize the core ideas from our earlier work [14], specifically leveraging historical information to predict missing values. In this approach, data segments before and after gaps are treated as independent time series. These univariate series are then transformed into multivariate formats, enabling bidirectional prediction using machine learning methods. The final imputation is derived by averaging these predictions, which further improve accuracy. As illustrated in Figure 1, the proposed framework consists of four stages:

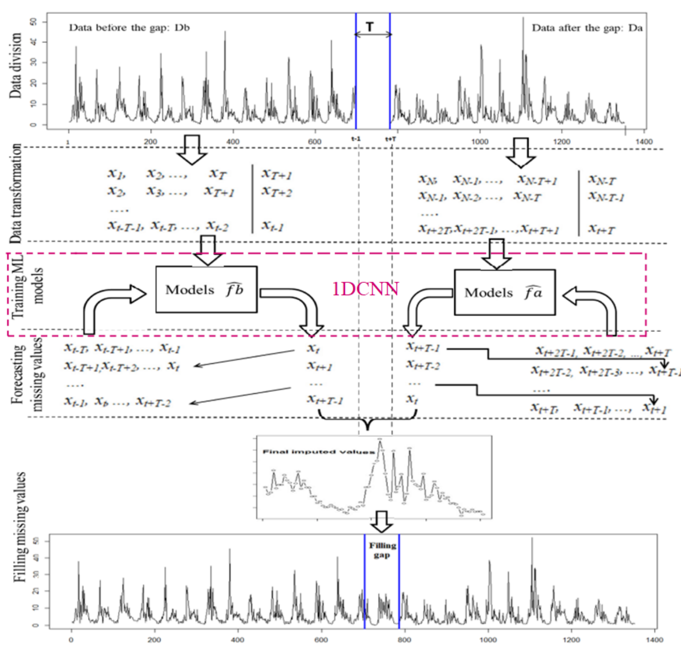

Fig. 1. Diagram of the approach to fill missing data [14]

Data division: For each gap of size T, we extract two separate time series from the original dataset: one representing the data before the gap (Db) and the other representing the data after the gap (Da). These two-time series are then treated as distinct and independent sequences. To address different missing data patterns, the proposed method employs a flexible approach. If the missing data occurs within the initial or final third of the time series, machine learning models are applied to the available data segments before or after the gap, respectively. For missing data located within the middle third of the series, the method utilizes both preceding and succeeding data segments for training and prediction.

Data transformation: To effectively capture temporal dependencies, the univariate time series is transformed into a multivariate format with T+1 dimensions, where T represents the gap size. This transformation enables the model to leverage historical data for accurate imputation.

Training Model: After converting the data into a multidimensional format, we primarily focus on applying the one-dimensional Convolutional Neural Network (1DCNN) for deep learning. To assess its effectiveness, we also compare it with traditional machine learning algorithms from previous research, including Random Forest, Support Vector Machine, K-nearest Neighbors, and Extra Trees.

Forecasting Missing Values: To impute missing values, we use a multi-step-ahead forecasting approach based on one-step-ahead predictions. For each missing segment (D_b or D_a), we iteratively predict T values using the previous T observed values as input. For example, as shown in Figure 1, data from $x(t-T)$ to $x(t-1)$ is used to estimate the missing value at $x(t)$. This process continues until all missing values are imputed. Imputation is conducted in both directions.

Filling missing values: To complete the missing data, the final imputed values are determined by averaging the corresponding elements from the forward and backward predicted vectors.

The following section describes in detail the algorithms used to estimate missing data

2.1 Proposed 1-dimensional Convolutional Neural Networks (1DCNN)

Convolutional Neural Networks (1DCNN) [11] are a type of feed-forward neural network that has excelled in processing 2D data such as images and video frames. To address the need for handling one-dimensional data, 1DCNN were developed. These networks offer several advantages over their 2D counterparts, including lower computational complexity, reduced parameter count, and the ability to operate on devices with limited resources. Consequently, 1DCNN architectures have been widely adopted in the literature. A detailed overview of 1DCNN, their applications, and recent advancements can be found in [8].

Figure 2 illustrates the architecture of the proposed one-dimensional convolutional neural network (1DCNN) designed for imputing missing values in univariate time series. This network is specifically engineered to handle variable-length input sequences, with each sequence length tailored to the corresponding missing data gap. The network features a sequential arrangement of layers, each contributing to the accurate prediction of missing data points. The network comprises the following layers:

Input Layer: The initial layer processes raw, one-dimensional time series data, accommodating varying input lengths based on the missing data gaps.

Convolutional Layers:

- `Conv1D (128, 3)`: The first convolutional layer employs 128 filters with a kernel size of 3 to extract local features from the input data.
- `Conv1D (64, 3)`: The second convolutional layer reduces the feature map size to 64 channels, refining the extracted features.
- `Conv1D (32, 3)`: The final convolutional layer further decreases the feature map size to 32 channels, capturing more abstract representations of the data.

Dense Layers:

- `Dense (128)`: This fully connected layer maps the output of the convolutional layers to a 128-dimensional space, enabling the capture of complex relationships between features.
- `Dense (1)`: The final dense layer produces a single output, representing the predicted value for the missing data point.

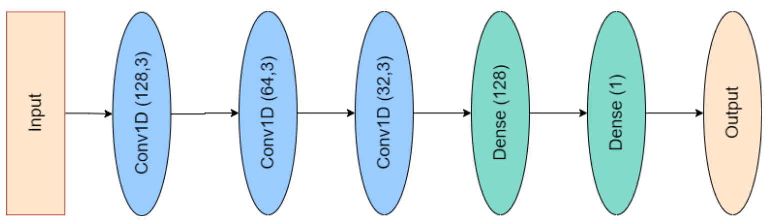

Fig. 2. Proposed 1DCNN architecture for imputing missing values

2.2 Machine Learning Models

Random Forest (RF): is a powerful ensemble learning method used for both classification and regression [1]. It operates by constructing multiple decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. To decrease correlation between individual trees, the algorithm uses a method called bagging, which involves creating bootstrap samples of the original data. Furthermore, when splitting a node during tree construction, a random subset of features is considered, rather than evaluating all features. This randomness helps to reduce overfitting and improve generalization.

Support Vector Machine (SVM): is another powerful algorithm widely used for both classification and regression tasks [2]. Their ability to effectively handle nonlinear relationships and high-dimensional data makes them particularly suitable for these problems. In the context of forecasting and imputation, which are regression problems, Support Vector Regression (SVR) is commonly employed. SVR aims to find an optimal hyperplane in a high-dimensional space that best fits the training data while minimizing prediction errors. This approach enables SVR to accurately capture complex patterns and generate reliable forecasts or imputed values.

K-Nearest Neighbors (KNN): is a non-parametric, instance-based learning algorithm renowned for its simplicity and effectiveness in certain contexts [3]. Unlike other machine learning models, KNN does not explicitly learn from the training data during a separate training phase. Instead, it defers computations until a prediction is required for a new data point. The algorithm operates by identifying the K closest data points to a given query point based on a chosen distance metric. For classification tasks, the predicted class is determined by the majority class among these K neighbors. In regression, the predicted value is the average of the target values of the K nearest neighbors. This flexibility makes KNN adaptable to various prediction problems.

Extra Trees (ET): is an ensemble learning algorithm that excels in both regression and classification tasks [5]. It is closely related to Random Forests but introduces additional randomness for efficiency and model diversity. Like Random Forests, Extra Trees construct multiple decision trees. However, they differ significantly in tree building. While Random Forests randomly select a subset of features at each node, Extra Trees consider all features and choose split points completely randomly. This extreme randomization reduces overfitting and often leads to comparable or even better performance than Random Forests with fewer trees. As a result, Extra Trees are computationally efficient and robust, making them suitable for various machine learning applications. Their combination of speed, accuracy, and adaptability positions them as a valuable tool in the machine learning

3 Experiments and Results

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

3.1 Data description

In this study, water level data from two hydrology stations along the Red River (the main river in the Northern Delta of Vietnam), specifically Vu Quang, Hanoi are utilized. The samples were recorded every three hours from January 1, 2008, to December 31, 2017, including 29224 samples. Figure 3 illustrates the data from the Vu Quang hydrology station.

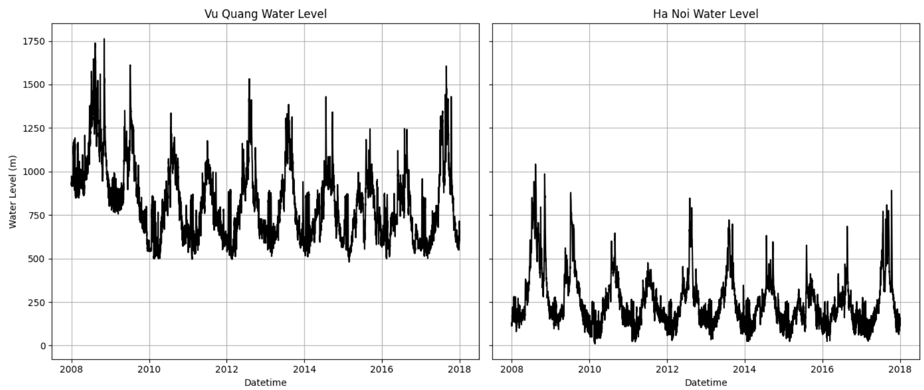

Fig. 3. Water level at Vu Quang and Hanoi hydrology stations of the Red River

3.2 Evaluation metrics

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

- Sim (Similarity): Measures how similar the imputed values (Y) are to the true values (X), assessing the overall accuracy of the imputation. The Sim value lies within the range $[0, 1]$, with values closer to 1 indicating a higher degree of similarity between the imputed values X and the true values Y. A Sim value of 1 represents perfect similarity, where the imputed values exactly match the true values, while a value closer to 0 signifies less similarity.

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

- MAE (Mean Absolute Error): is calculated as the average of the absolute differences between predicted values and the actual values. Mathematically, it is defined as:

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

A lower MAE value indicates that the predictions are closer to the actual values, suggesting a more accurate imputation model or predictive model.

- RMSE (Root Mean Squared Error): is the square root of the average of the squared differences between the predicted values and the actual observed values. A lower RMSE value indicates that the model's predictions are closer to the actual values, implying a more accurate imputation model. RMSE is defined as:

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

- FSD (Fraction of Standard Deviation): compares the variability in the imputed values to the variability in the actual values. The FSD is calculated as the ratio of the standard deviation of the errors (i.e., the differences between the actual and imputed values) to the standard deviation of the actual values. FSD can be expressed as:

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

- NSE (Nash-Sutcliffe Efficiency): The NSE metric is commonly used to assess the accuracy of hydrological models. It ranges from $-\infty$ to 1, where values closer to 1 signify a better performance between the observed and predicted water levels [12].

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

3.3 Experiment results

This section presents the experimental results obtained from the proposed approach and various imputation methods. Since the actual values are missing, a direct assessment of the imputation methods' performance is not feasible. To address this, artificial missing data was introduced into the complete time series, enabling the evaluation of these imputation methods. This study considers five different missing data rates across two univariate time series. For each rate, gaps are randomly generated at 10 different positions throughout the full series. The final results represent the average performance across these 10 gaps. For each dataset, the imputation process is repeated 50 times.

The 1DCNN model is optimized with key hyperparameters: the number of epochs is set to 200, ensuring the model has sufficient time to learn from the data. Although 200 epochs are used, the model is equipped with an early stopping mechanism with a patience value of 30, allowing training to stop if no improvement is observed after 30 epochs, this helps balance optimal performance and overfitting. The batch size is configured as 16, balancing training speed and model stability. Finally, the validation split is set to 0.2, ensuring that 20% of the training data is used to evaluate the model's performance during training.

Table 1 presents a comparative analysis of various machine learning methods used to estimate missing data in two time series: Vu Quang and Hanoi. The performance of these methods is evaluated using metrics such as Sim, MAE, RMSE, FSD, and NSE.

The 1DCNN model consistently outperforms other methods across all missing data lengths and locations based on MAE, RMSE, and Sim metrics. It achieves the lowest MAE, RMSE, and FSD values, alongside the highest Sim and NSE values, underscoring its accuracy and ability to capture patterns effectively. This robustness and adaptability to different data conditions are particularly evident as the length of missing data increases, where the performance gap between CNN and other methods becomes more pronounced. While other methods maintain reasonable performance for shorter missing data lengths, their accuracy deteriorates rapidly as the missing period extends, highlighting their limitations in capturing long-term dependencies. Specifically, both SVM variants (SVM-RBF and SVM-Linear) exhibit inconsistent performance across different scenarios, possibly due to their sensitivity to hyperparameter tuning and kernel selection. Similarly, KNN struggles significantly with longer missing data periods, reflecting its reliance on nearby data points for accurate imputation. Let's examine the performance of each method in detail:

1DCNN model: In Vu Quang, the 1DCNN method demonstrates the highest performance, with Sim values ranging from 0.81 to 0.82. It also achieves the lowest MAE and RMSE values, between 14.47 to 74.62, and 17.51 to 74.62, respectively. Furthermore, the FSD is the lowest across various missing data lengths, and the NSE value is superior to that of other methods. Likewise, in Hanoi, 1DCNN consistently delivers the best results across all metrics, with Sim values ranging from 0.82 to 0.85. Both MAE and RMSE remain the lowest, further proving 1DCNN's effectiveness in handling missing data gaps, especially as the length of these gaps increases.

Random Forest: In Vu Quang, the Random Forest method shows decent performance but still lags behind CNN, with Sim values ranging from 0.78 to 0.81. The MAE and RMSE values are higher than those of 1DCNN. A similar pattern is observed in Hanoi, where RF exhibits a decline in performance as the length of the missing data increases. This trend suggests that RF is less effective than CNN in dealing with longer data gaps.

Support Vector Machine: Both SVM (RBF) and SVM (Linear) demonstrate significant variability in performance across both time series. Their Sim values range from 0.73 to 0.81, and their error metrics are higher than those of 1DCNN and RF. Notably, SVM's performance deteriorates substantially as the length of the missing data increases, indicating its limitations under these conditions.

Extra Trees: The ET method exhibits performance comparable to RF but still falls short of 1DCNN. With Sim values ranging from 0.76 to 0.82, ET shows relatively better predictive ability than SVM. However, it struggles to match 1DCNN's effectiveness, particularly when faced with large missing data gaps.

K-nearest Neighbors: In Vu Quang, KNN performs the worst among the evaluated methods, with Sim values ranging from 0.72 to 0.79 and the highest error metrics. This poor performance suggests that KNN is unsuitable for handling long missing data gaps. A similar outcome is observed in Hanoi, where KNN's results are significantly worse than those of 1DCNN and RF. This performance reflects KNN's strong dependence on nearby data points and its difficulty in making accurate predictions when there are substantial missing data gaps.

Table 1. Performance of ML methods for estimating missing data in Vu Quang and Hanoi time series

Missing	Methods	Sim	MAE	RMSE	FSD	NSE	Sim	MAE	RMSE	FSD	NSE
		Vu Quang					**Hanoi**				
12h	**1DCNN**	**0.82**	**14.47**	**17.51**	**0.84**	**0.21**	**0.82**	**15.25**	**17.59**	**0.86**	**-0.16**
	RF	0.81	16.27	18.86	0.98	0.12	0.75	19.88	22.71	1.05	-1.17
	SVM (RBF)	0.78	17.22	19.97	1.14	-0.71	0.74	22.96	25.74	1.22	-2.61
	SVM (Linear)	0.79	16.61	19.32	1.1	-0.28	0.75	21.37	24.13	1.19	-2.02
	ET	0.81	16.39	19.04	0.94	0.09	0.76	20.5	23.97	0.96	-1.02
	KNN	0.79	17.39	20.44	0.85	-0.26	0.74	21.5	24.92	0.89	-2.96
24h	**1DCNN**	**0.82**	**19.41**	**22.76**	0.98	**-0.1**	**0.83**	**19.07**	**22.64**	**0.8**	**0.07**
	RF	0.78	24.58	28.26	1.04	-1.14	0.77	31.8	36.7	1.13	-2.37
	SVM (RBF)	0.8	21.47	25.44	1.13	-0.68	0.81	31.25	36.17	1.1	-0.08
	SVM (Linear)	0.79	23	26.68	1.28	-0.64	0.8	31.8	36.39	1.09	-0.23
	ET	0.77	24.89	29.15	0.98	-1.9	0.78	31.95	37.27	1.02	-1.13
	KNN	0.72	34.37	39.66	**0.64**	-3.49	0.8	29.88	34.63	0.91	-0.22
48h	**1DCNN**	**0.81**	**31.33**	37.92	0.71	-0.65	**0.82**	**24.58**	**29.61**	0.86	**-0.39**
	RF	0.8	34.62	40.34	1.18	-0.83	0.81	26.63	31.8	1.09	-0.46
	SVM (RBF)	0.77	37.94	43.27	0.9	-1.92	0.81	27.03	31.57	0.93	-0.55
	SVM (Linear)	0.8	32.93	**37.88**	1.18	**-0.48**	0.79	31.43	35.97	1.11	-1.6
	ET	0.79	35.27	40.97	1.18	-0.7	0.81	26.65	31.27	0.99	-0.42
	KNN	0.77	42.22	47.72	**0.49**	-1.45	0.8	30.89	35.88	**0.64**	-1.09
72h	**1DCNN**	**0.81**	**42.9**	**50.98**	0.92	**-0.35**	**0.85**	**20.73**	**25.61**	**0.52**	**-0.19**
	RF	0.8	46.75	56.53	1.22	-0.65	0.8	29.56	34.16	1.07	-0.97
	SVM (RBF)	0.75	62.49	74.28	0.72	-3.51	0.74	42.24	48.45	0.57	-3.17
	SVM (Linear)	**0.81**	47.65	57.63	1.25	-0.56	0.81	27.34	32.55	0.94	-0.74
	ET	0.8	47.6	58.02	1.23	-1.31	0.82	26.18	31.64	0.94	-0.76
	KNN	0.74	75.19	87.78	**0.67**	-2.66	0.72	49.38	56.98	0.56	-6.29
120h	**1DCNN**	**0.81**	**62.75**	**74.62**	0.68	-0.54	**0.83**	**35.43**	**43.52**	0.59	-0.46
	RF	0.8	64.63	76.08	1.38	-0.58	**0.83**	38.55	46.63	1.16	**-0.37**
	SVM (RBF)	0.73	103.24	122.72	0.56	-5.34	0.76	56.63	66.1	0.77	-4.49
	SVM (Linear)	0.8	66.53	76.15	1.27	**-0.4**	0.76	123.92	163.4	1.29	-178.12
	ET	0.79	73.71	86.52	1.01	-1.05	0.82	42.31	50.26	0.93	-0.63
	KNN	0.78	80.01	93.97	**0.3**	-1.4	0.8	42.62	51.99	**0.58**	-2.64

In addition to quantitative analysis, data visualization offers valuable insights. Figure 4 compares true values and imputed values from the 1DCNN model at missing position 1 in GAP 24, with indices representing the corresponding missing hours. The results indicate that the 1DCNN model performs well with high imputation accuracy, though there are notable discrepancies at inflection points in the time series, where predicted values diverge significantly from the ground truth. The model successfully captures the overall trend of the data, suggesting it has learned the underlying patterns of the time series.

1DCNN demonstrate superior performance in time series imputation due to several key factors. Firstly, unlike traditional ML methods that rely on manual feature engineering, 1DCNN excel at automatically learning relevant features directly from the data. This is particularly advantageous in time series analysis where intricate patterns, such as seasonality, trends, and cyclic behaviors, are often difficult to capture manually. Secondly, 1DCNN are inherently designed to

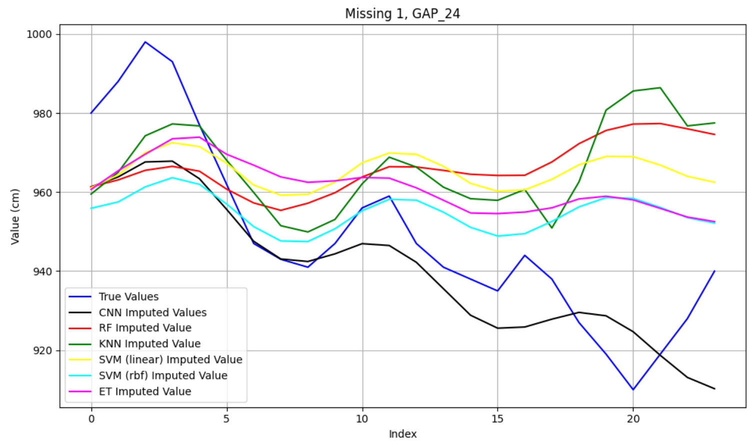

Fig. 4. Visualization of imputed values generated by different methods at missing position 1, gap size 24 on Vu Quang dataset.

process sequential data, making them well-suited for time series analysis. The convolutional layers in 1DCNN effectively capture local dependencies and temporal patterns within the data. Thirdly, 1DCNN possess strong generalization capabilities, allowing them to adapt to diverse time series characteristics and handle varying lengths of missing data. Finally, the hierarchical structure of 1DCNN enables the model to learn complex representations of the data, leading to improved accuracy in imputing missing values.

4 Conclusion

This study delves into the application of machine learning and deep learning algorithms for the critical task of imputing missing data in time series. By transforming univariate data into a multivariate format and capitalizing on both pre-and post-gap temporal patterns, we develop a robust framework for multi-step-ahead prediction. Our proposed lightweight 1DCNN model consistently outperforms traditional methods across various missing data lengths in the Vu Quang and Hanoi datasets. This study highlights the efficacy of deep learning in addressing the challenges posed by missing data in time series analysis. Future work will explore the integration of attention mechanisms within the CNN architecture to further enhance imputation accuracy and adaptability.

References

1. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)
2. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning **20**(3), 273–297 (1995)

3. Cover, T., Hart, P.: The nearest neighbor decision rule. IEEE Transactions on Information Theory (1967)
4. Eum, Y., Yoo, E.H.: Imputation of missing time-activity data with long-term gaps: A multi-scale residual cnn-lstm network model. Computers, Environment and Urban Systems **95**, 101823 (2022)
5. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine learning **63**(1), 3–42 (2006)
6. Jadhav, A., Pramod, D., Ramanathan, K.: Comparison of performance of data imputation methods for numeric dataset. Applied Artificial Intelligence **33**(10), 913–933 (2019)
7. Kim, J., Oh, S., Kim, H., Choi, W.: Tutorial on time series prediction using 1d-cnn and bilstm: A case example of peak electricity demand and system marginal price prediction. Engineering Applications of Artificial Intelligence **126**, 106817 (2023)
8. Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., Inman, D.J.: 1d convolutional neural networks and applications: A survey. Mechanical systems and signal processing **151**, 107398 (2021)
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. Mallinson, H., Gammerman, A.: Imputation using support vector machines. University of London Egham, UK: Department of Computer Science Royal Holloway (2003)
11. Montesinos López, O.A., Montesinos López, A., Crossa, J.: Convolutional Neural Networks, pp. 533–577. Springer International Publishing, Cham (2022)
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. Shaadan, N., Rahim, N.: Imputation analysis for time series air quality (pm10) data set: A comparison of several methods. In: Journal of Physics: Conference Series. vol. 1366, p. 012107. IOP Publishing (2019)
16. 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)
17. 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)
18. 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)
19. Zeng, Z., Kaur, R., Siddagangappa, S., Rahimi, S., Balch, T., Veloso, M.: Financial time series forecasting using cnn and transformer. arXiv preprint arXiv:2304.04912 (2023)
20. Zhao, B., Lu, H., Chen, S., Liu, J., Wu, D.: Convolutional neural networks for time series classification. Journal of systems engineering and electronics **28**(1), 162–169 (2017)