WBDI Approach for Univariate Time Series Imputation

Thi-Thu-Hong Phan, Ngoc-Huy Dao, Quang-Minh Doan, Thieu-Quang Dinh, Quan-Bao Nguyen
Artificial Intelligence Department, FPT University, Da Nang, Vietnam
E-mail: hongptt11@fe.edu.vn, huydnde160024@fpt.edu.vn, minhdqde170407@fpt.edu.vn,
quangdtde160066@fpt.edu.vn, baonqde170736@fpt.edu.vn

Keywords: missing data, weighted bi-directional imputation, univariate time series, ensemble learning methods, AdaBoost

Received: June 1, 2024

Incomplete data can significantly impact the results and reduce data value for machine learning systems. Simple imputation methods often fail to capture the intricate patterns and relationships within time series data, leading to inaccurate analysis. This study proposes a novel approach called "weighted bi-directional imputation, WBDI" to address this challenge in univariate time series data. The proposed approach leverages machine learning models and utilizes data before and after the missing segment, incorporating weights to prioritize relevant information. To evaluate its effectiveness, experiments are conducted using eleven machine learning algorithms on three real-world datasets (Phu Lien humidity, Cua Ong climate, and Ha Noi water level) with varying sizes and sampling frequencies. We evaluate performance using Similarity, Normalized Mean Absolute Error (NMAE), Root Mean Square Error (RMSE), Fraction of Standard Deviation (FSD), and Fractional Bias (FB). The results demonstrate that ensemble learning methods generally outperform other approaches. Notably, the AdaBoost method consistently performs well across all datasets and evaluation metrics. Its exceptional performance is highlighted by achieving a Similarity of 0.99 when completing one-day missing data on the Hanoi water level, illustrating its high reliability and accuracy in imputing missing values.

Povzetek: Raziskava uvaja metodo tehtane dvosmerne imputacije (WBDI) za izpolnjevanje manjkajočih vrednosti v univariatnih časovnih vrstah. Metoda združuje strojno učenje in podatke pred ter po manjkajočih segmentih, kar omogoča odlične rezultate z uporabo AdaBoosta (npr. 99% o vodostaju v Hanoju).

1 Introduction

Time series data are ubiquitous in many domains in our life such as health care [29], economics [20], meteorology [9], finance [4], astronomy [1]. However, a common challenge in analyzing such data is the presence of missing values caused by factors like malfunctioning sensors, transmission issues, or inaccurate calculations. Analyzing data with missing values can lead to biased conclusions and inaccurate predictions and it is more challenging to find significant patterns and relationships in the data. Therefore, imputation techniques are essential as a standard method of handling missing [21].

Time series data can be categorized into univariate and multivariate data, each requiring specific techniques to handle missing values. For univariate time series, various imputation techniques have been developed, including classical statistical methods. Some methods are mean [2], median, and mode interpolation, which are straightforward but may not always capture the temporal dynamics of the data adequately. One notable method that offers an improvement over earlier techniques is the Last Observation Carried Forward (LOCF). This approach assigns the last observed value to the missing data points, leveraging the relationship between a current observation at a point in time, t_n and its predecessor at t_{n-1} [24]. This technique is particularly useful when the assumption that the most recent observation is a good estimate for the missing value. Linear and spline interpolation techniques are also widely employed for their computational efficiency and ability to handle short, missing gaps. These methods assume a smooth trend between the observed data points, making them suitable for datasets with consistent and regular sampling. However, these techniques often oversimplify the underlying temporal patterns and struggle with irregularly sampled data or large missing gaps, limiting their effectiveness in more complex scenarios [24].

The Autoregressive Integrated Moving Average (ARIMA) model is highly effective for time series data that exhibit a clear trend and seasonality. It is widely used for short to medium-term forecasting in univariate time series data. However, ARIMA can struggle with complex relationships or irregularly sampled data and requires the data to be linear to perform optimally [3]. The Kalman filter is another powerful method for imputing missing values, particularly useful in contexts with noisy measurements. For example, in [8], the Kalman filter was utilized for imputing LTE (Long Term Evolution) spectrum data, which is characterized by high seasonality and a univariate nature. The filter's strength lies in its ability to

handle noisy data effectively and provide robust estimates. Another technique, the Dynamic Time Warping-Based Imputation (DTWBI), was employed to estimate large missing values in univariate time series, as described in [27]. DTWBI is particularly useful under the assumption that missing values exist within the series and can leverage the alignment of time series to fill in gaps. An extension of this method, eDTWBI, was introduced in [26]. This enhanced version considers both the data before and after the gap to identify similar patterns more accurately, leading to improved imputation performance.

In addition to applying statistical methods and finding similar values to replace incomplete values, estimating missing values based on machine learning has emerged as an alternative direction. This has become a growing area of research, with studies demonstrating its effectiveness. For instance, a study by Han et al. (2023) integrated Support Vector Regression with decomposition methods to recover data in a wastewater treatment process [19]. Another study compared Support Vector Machines (SVM) and Random Forests (RF) for forecasting missing data in time series [25]. The authors concluded that machine learning methods are noteworthy approaches for imputation tasks. Furthermore, Fang et al. (2020) explored advanced deep learning techniques like GRU-D, GRUI-GAN, and E^{2}GAN for imputing missing values in time series data [15]. Du et al. introduced a novel deep-learning model to impute missing values by leveraging a self-attention mechanism and joint-optimization training approach. This method demonstrates its potential for addressing missing data challenges in imputation accuracy and training speed, particularly for multivariate time series data [13]. Kazijevs et al. (2023) [22] conducted a comprehensive survey on the imputation of missing values in multivariate time series data. Notably, the authors adeptly addressed the limitations of existing studies in this field by conducting data-centric experiments on health datasets, shedding light on effective strategies for handling missing data in complex temporal contexts. Emmanuel et al. (2021) carried out an exhaustive examination of machine learning techniques for handling missing data. Their survey covered patterns and mechanisms of missingness, evaluation metrics, and various approaches for handling missing data [14]. They assessed the performance of these techniques on various datasets and identified noise as a negative factor impacting imputation methods. However, the study acknowledged limitations: the absence of nonlinear terms in the models and the high computational cost associated with the most effective methods.

To address the limitations, this study investigates a new framework for imputing missing values in univariate time series data. The proposed approach, named weighted bi-directional imputation (WBDI), utilizes machine learning models to capture complex patterns and relationships within the data. WBDI leverages data from both before and after the missing segment, incorporating weights to prioritize relevant information. This combined approach aims to improve the learning process and enhance the accuracy of imputation.

The structure of the paper is organized as follows: In Section 2, we provide a detailed description of the methods applied in this study. Section 3 outlines the experiments conducted. The presentation of results and subsequent discussion are presented in Section 4. Some concluding remarks are given in Section 5

2 Methodology

In this section, we present our proposed approach for imputing missing values in univariate time series by combining the forecasting results before and after the studied gap, simultaneously considering weights for each part of the data. We then briefly introduce the machine learning algorithms used for implementing the proposal.

2.1 The proposed approach, WBDI

The central concept explored in this article involves applying machine learning techniques to address the issue of missing data. The proposed approach, termed weighted bi-directional imputation (WBDI), encompasses utilizing the available data before and after the missing segments to impute the missing values considering weights for each part of the available data. This makes it possible to enhance the learning database and the size of learning data so it can improve the performance of the imputation process.

The methodology comprises four sequential steps, as illustrated in Fig. 2. Firstly, for each gap, the data preceding and succeeding this gap are considered two learning datasets. Then these univariate data are transformed into multivariate data by choosing an appropriate window size. Subsequently, each machine learning model is trained on the forward and backward data corresponding to the missing position. Many ML methods are employed to predict the absent values. Finally, the outcome is derived by amalgamating the predictions from both models based on the weights assigned to each part of the results. The details are presented as follows:

Step 1 - Transforming data: The univariate data are split into two datasets: the available data before the gap, denoted D_{before}, and the available data after the gap, namely D_{after}. At this step, by determining the suitable window size N, the univariate datasets are converted into N-dimensional data. This indicates that the next value in the time series is predicted using the N previous values, and the preceding value is estimated using the N subsequent values. In particular, the training set and output vector with the X_{before} database are forward direction from x_1 (see Fig. 1). As illustrated in Fig. 1, the training set and output vector for the X_{before} data are backwards set up from x_d.

Step 2 - Training model: At this stage, the machine learning algorithms are trained on the dataset (X_{before}), called f_{before}, and on the dataset (X_{after}), called f_{after}.

Table 1: Method overview

Paper	Dataset	Model	Metric	Key finding
Moritz et al. (2015)	Airpass Beersales Google SP dataset	LOCF Linear and Spline interpolation	RMSE MAPE	LOCF imputes missing values in stable time series but fails with seasonality or trends. Linear and spline interpolation performs better, with spline handling complex non-linear patterns.
Chaudhry et al. (2019)	LTE spectrum	Kalman filter	MAPE, Standard Error of Mean	Converting highly seasonal univariate data to a multivariate form and using MICE significantly improves imputation and prediction accuracy, outperforming traditional methods like Kalman filtering.
Phan et al. (2020)	CO2 concentrations, Phu Lien humidity/temperature, Cua Ong air temperature	DTWBI, eDTWBI	Similarity, NMAE, RMSE, FSD, FB	Both methods outperform traditional imputation techniques; eDTWBI particularly enhances accuracy and preserves frequency components in seasonal data with extensive missing periods
Phan et al. (2020)	CO2 concentrations, Phu Lien air temperature, NNGC, Ba Tri temperature	SVM, RF	Similarity, NMAE, RMSE, FSD, FB	Both SVM and RF-based imputation methods show significant improvements in accuracy for imputing missing values in univariate time series data, outperforming other techniques
Fang, Wang C. (2020)	PhysioNet datasets (MIMIC-III, KDD)	GRU-D, GRUI-GAN, E2GAN	MSE, MAE, RMSE, AUC	Each model offers unique approaches to handle missing data in time series, with E2GAN achieving state-of-the-art performance in imputation tasks
Wenjie Du (2022)	PhysioNet-2012 Air-Quality Electricity	Self-attention	MAE RMSE MRE	The SAITS model outperformed existing imputation methods, showing lower error rates in filling missing values in time-series data
Han et al. (2023)	WWTP	SVR	RMSE SIM	UIM surpasses seven methods for imputing WWTP missing values, yielding better RMSE, similarity, and efficiency. Suitable for nonlinear, nonstationary data.

$$D_{before} = \{x_1, x_2, \ldots, x_{t-1}\} \Longrightarrow X_{before} = \begin{bmatrix} x_1 & x_2 & \cdots & x_N \\ x_2 & x_3 & \cdots & x_{N+1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{t-N-1} & x_{t-N} & \cdots & x_{t-2} \end{bmatrix} \begin{bmatrix} x_{N+1} \\ x_{N+2} \\ \vdots \\ x_{t-1} \end{bmatrix}$$

$$D_{after} = \{x_{t+T-1}, x_{t-T-2}, \ldots, x_d\} \Longrightarrow X_{after} = \begin{bmatrix} x_d & x_{d-1} & \ldots & x_{d-N+1} \\ x_{d-1} & x_{d-2} & \cdots & x_{d-N} \\ \vdots & \vdots & \ddots & \vdots \\ x_{t-T+N+1} & x_{t+T+N} & \ldots & x_{t+T+2} \end{bmatrix} \begin{bmatrix} x_{d-N} \\ x_{d-N-1} \\ \vdots \\ x_{t-T-1} \end{bmatrix}$$

Figure 1: Data transformation

In this paper, we conduct eleven well-known machine learning models to estimate missing values.

Step 3 - Forecasting missing data: To predict T absent values, the proposed model involves applying T times the process of one-step-ahead prediction. The final one indicates that an output value will be forecasted based on a given set of observed data. The predicted value at each step appending to the original data will be used to estimate the next outcome.

Predicting the T values for the (X_{before}) data is explained in detail as follows:

- i) Using ML models to predict the missing value at time t based on N previously real data points.
- ii) Using ML models to forecast the value at time $t+1$ based on N historical values in which $N-1$ are real values of the original series from $t-N+1$ to $t-1$, and the predicted value at time point t
- ...
- T) Using ML models to forecast the value at time $t+T$ based on N past values in which $N-T$ are actual values of the real series from $t-N+T$ to $t-1$, and T previously predicted values. Finally, we get Y_{before}. For data X_{after}, we repeat from step 1 to step T to obtain the estimated values Y_{after}.

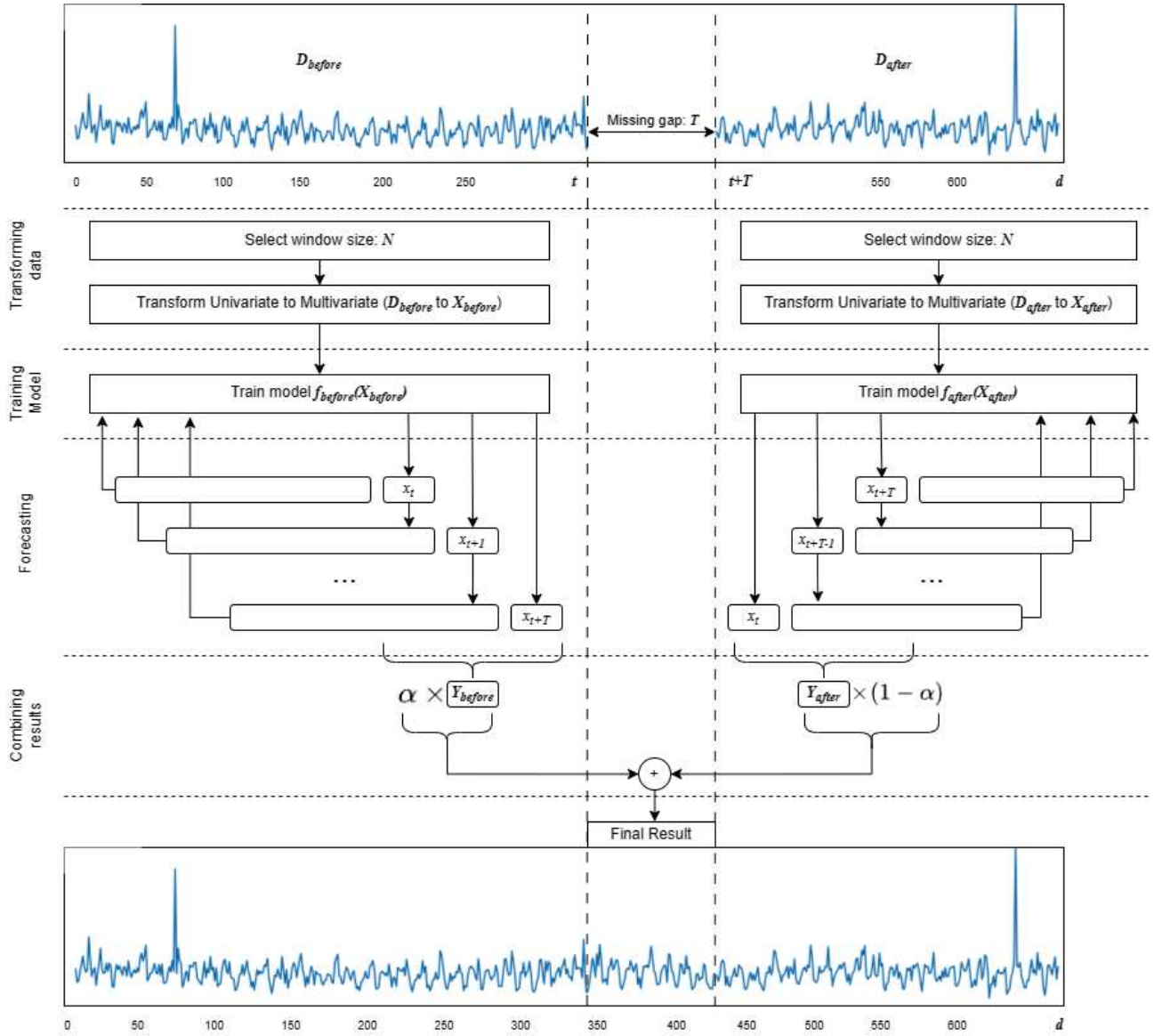

Figure 2: Process of imputing on univariate time series

Step 4 - Combining results: After both forecasting processes are finished, the two predicted values will be combined taking into account the contribution of the learning dataset size, namely (α) ratio. The α coefficient represents the proportion of the size of the data. It is determined by the relative sizes of the datasets from before and after the missing gap. The rationale behind using dataset length as a weighting factor is that longer datasets generally provide more reliable information for prediction. If one dataset is larger than the other, it will have a greater contribution to the final result. This alpha coefficient represents the ratio between preceding and following data shown in Formula 1. This coefficient takes values between 0 and 1, the larger the value, the more it affects the final result.

$$\alpha = \frac{len(D_{before})}{len(D_{before}) + len(D_{after})} \quad (1)$$

Finally, the result is calculated as follows:

$$FinalResult = \alpha \times Y_{before} + (1-\alpha) \times Y_{after} \quad (2)$$

2.2 Machine learning methods

This paper investigates the performance of machine learning methods for predicting missing data in the time series. A variety of machine learning algorithms are employed, ranging from simple linear models like Linear Regression to complex ensemble methods. Linear Regression serves as a baseline for comparison, while algorithms such as KNN, SVM, and Decision Trees were selected due to their ability to model non-linear patterns within the data. Ensemble methods, including Random Forest, Extra Trees, Gradient Boosting, AdaBoost, and XGBoost, are incorporated to potentially enhance predictive performance through the

combination of multiple models. A summary of these algorithms is presented in this section.

2.2.1 Linear Regression (LR)

LR is a statistical method used for modeling the relationship between a dependent variable and one or more independent variables. The fundamental idea behind linear regression is to find the best-fitting linear relationship (a straight line) that minimizes the sum of the squared differences between observed and predicted values. This method is widely used for prediction, understanding relationships between variables, and identifying the strength and direction of those relationships [28].

The model assumes that the relationship between the variables can be represented by a linear equation of the form:

$$y = \beta_0 + \beta_1 \times x_1 + \beta_2 \times x_2 + ... + \beta_n \times x_n + \epsilon \quad (3)$$

where y is the dependent variable, x_1, x_2, ..., x_n are the independent variables, β_0 is the intercept, $\beta 1$, $\beta 2$, ..., βn are the coefficients representing the relationship's slope, ϵ is the error term, representing the unobserved factors affecting the dependent variable.

The coefficients (β values) are estimated using ordinary least squares, which minimizes the sum of squared differences between the observed and predicted values.

2.2.2 K-Nearest Neighbor (KNN)

KNN is a supervised machine learning algorithm used for classification and regression tasks. KNN is a non-parametric and lazy learning algorithm, meaning it doesn't make assumptions about the underlying data distribution and it postpones the actual learning until a prediction is needed. The performance of KNN can be influenced by the choice of distance metric, the value of K, and the nature of the data.

For classification tasks, to predict the class of a new data point, the algorithm identifies K nearest data points in the training set based on a distance metric (commonly Euclidean distance). As illustrated in Fig.3, the predicted class is then determined by the majority class among these K neighbors.

For regression tasks, to predict the target value for a new data point, the algorithm determines K nearest data points in the training set. After that, the predicted value is computed as the average (or weighted average) of the target values of these K neighbors [23].

2.2.3 Support Vector Machine (SVM)

SVM is a popular supervised machine learning algorithm used for both classification and regression tasks. In the context of classification, as shown in Fig. 4, SVM aims to find a hyperplane that best separates the data into different classes while maximizing the margin between classes.

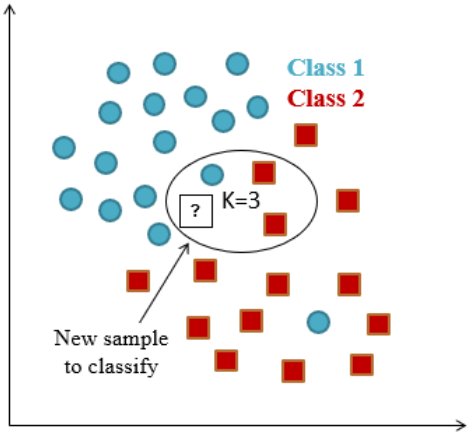

Figure 3: K-Nearest Neighbor algorithm

The margin is the distance between the hyperplane and the nearest data points of each class.

For the regression task, the goal is to predict a continuous output variable, SVM shares the basic principles of SVM for prediction of real-valued quantities. It aims to find a hyperplane to capture the underlying trend in the data rather than separating classes.

To address scenarios where the relationship between input features and the target variable is non-linear, SVM employs kernel functions. These functions facilitate the mapping of input data into a higher-dimensional space, enabling the algorithm to capture intricate non-linear relationships that may exist in the underlying data. SVM proves particularly advantageous in situations characterized by non-linearities or when dealing with noisy data, showcasing its versatility in handling complex real-world regression problems [11].

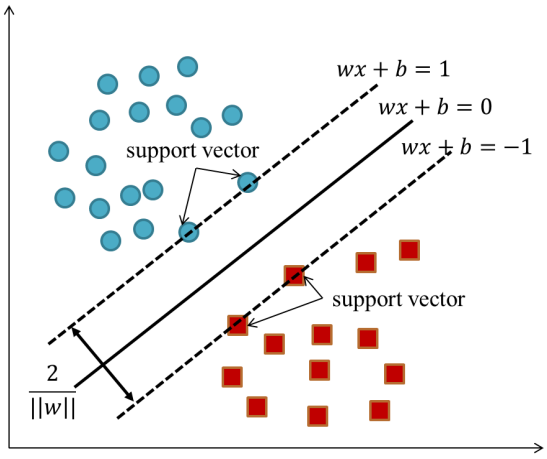

Figure 4: Support Vector Machine

2.2.4 Decision Tree (DT)

DT is known for its interpretability and ease of visualization. This method is capable of handling both numerical

and categorical data and can automatically handle feature selection. It works by recursively partitioning the dataset into subsets based on the values of input features. The goal is to create a tree-like structure where each internal node represents a decision based on a particular feature, each branch represents the outcome of that decision, and each leaf node represents the final prediction or decision (Fig. 5). However, it is vulnerable to overfit, especially when the tree is deep and captures noise in the training data. Techniques like pruning and setting a minimum number of samples per leaf are often used to mitigate overfitting [5].

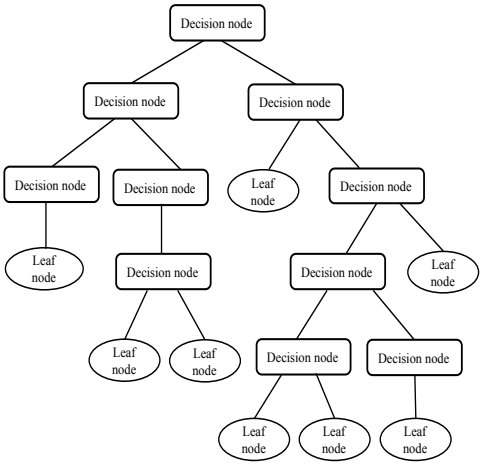

Figure 5: Decision Tree algorithm

voting mechanism (for classification) or averaging (for regression) to improve overall predictive accuracy and generalization to new, unseen data. The randomness introduced in the tree-building process helps reduce overfitting and enhances the robustness and effectiveness of the model. Fig. 6 explains the main point of RF.

The key features of the Random Forest algorithm encompass its capability to handle high-dimensional datasets, provide assessments of feature significance, and maintain good performance without extensive hyperparameter tuning. It is a powerful and widely used algorithm in machine learning, known for its versatility and effectiveness in various applications.

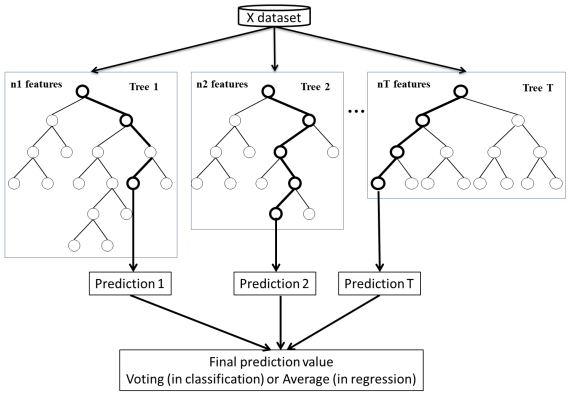

Figure 6: Random Forest algorithm

2.2.5 Bagging

Bagging which stands for Bootstrap Aggregating, is an ensemble learning technique designed to improve the stability and accuracy of machine learning algorithms. It involves training multiple instances of a base learning algorithm on different subsets of the training data and then combining their predictions. The process of creating these subsets involves random sampling with replacement, a method known as bootstrap sampling.

The key idea behind Bagging is that by training multiple models on diverse subsets of the data and combining their predictions, the ensemble model tends to generalize better and be more robust to variations in the training data. One of the most well-known implementations of Bagging is the Random Forest algorithm, which employs bagging with decision trees as the base models.

The benefits of Bagging include reducing overfitting, improving model stability, and enhancing predictive performance, especially when dealing with complex or noisy datasets [6].

2.2.6 Random Forest (RF)

RF is an ensemble learning method that builds multiple decision trees using bootstrapped samples of the training data and a random subset of features at each split [7]. The algorithm combines the predictions of these trees through a

2.2.7 Extra Tree (ET)

ET is an ensemble learning algorithm that builds multiple decision trees during training, and for each split in the trees, it randomly selects the feature to split on and chooses the splitting threshold without searching for the optimal values. This introduces extra randomness compared to Random Forests, leading to a diverse set of trees that collectively make predictions for classification or regression tasks. The final prediction is typically determined by averaging (for regression) or voting (for classification) over the predictions of individual trees in the ensemble.

The extra randomness in ET can lead to a reduction in variance, making it less prone to overfitting on the training data. ETs can be particularly useful in scenarios where a diverse set of weak learners (trees) is desired to create a robust and accurate ensemble model [18].

2.2.8 AdaBoost

AdaBoost is an ensemble learning method that focuses on improving the performance of weak learners by giving more weight to misclassified instances. It iteratively combines weak models, emphasizing the areas where they struggle. In the context of time series imputation, AdaBoost can be effective in capturing and improving imputation accuracy for specific temporal patterns that may be challenging for other algorithms. Its adaptive nature makes it partic-

ularly useful when dealing with complex and dynamic time series data [16].

In various problem domains, AdaBoost emerges as a versatile tool. By adeptly capturing and improving imputation accuracy for specific temporal patterns, AdaBoost contributes to enhancing the overall accuracy of the final predictions.

2.2.9 Gradient Boosting (GB)

GB is a method employed in the development of predictive models, predominantly applied in regression and classification tasks. This technique often represents prediction models through decision trees to determine the most accurate predictions. Similar to other boosting methods, Gradient Boosting builds multiple weak learners sequentially, with each one correcting the errors of its predecessor. It uses a gradient descent optimization process to minimize the loss function.

In time series imputation, Gradient Boosting can learn and adapt to the temporal patterns, iteratively improving the imputation accuracy. The sequential nature of boosting allows the model to focus on challenging instances, making it robust in handling missing values across various time points [17].

Furthermore, the iterative nature of Gradient Boosting facilitates nuanced learning from the sequential nature of time series data, enabling the model to capture evolving patterns over time. This adaptability enhances its performance in scenarios where temporal dependencies play a crucial role in the imputation process. The gradient descent optimization ensures a systematic refinement of the model, optimizing its predictive capabilities across varying temporal contexts.

2.2.10 XGBoost(XGB)

XGB is an optimized and efficient implementation of gradient boosting that includes regularization and parallel processing, making it highly effective for various machine learning tasks. In time series imputation, XGBoost extends the principles of gradient boosting to handle missing data points efficiently. Its regularization techniques contribute to preventing overfitting, while parallel processing enhances computational efficiency, making XGBoost a powerful choice for accurate and scalable imputations in time series data [10].

The robustness of XGBoost in handling time series data is underscored by its ability to capture intricate temporal dependencies, enabling the model to discern evolving patterns over consecutive time points.

2.2.11 Voting

Voting is an ensemble method that combines the predictions of multiple models (e.g., classifiers or regressors) and selects the most common prediction for classification or averages the predictions for regression. In time series imputation, a voting ensemble can provide a robust and balanced approach by aggregating the predictions of different algorithms. This approach leverages the diversity of models to handle various temporal patterns, contributing to a more reliable and accurate imputation process [12].

The versatility of a voting ensemble extends to its adaptability in handling dynamic time series data, where the temporal evolution of patterns requires a nuanced approach. By leveraging the collective insights from multiple algorithms, the ensemble method provides a comprehensive perspective, capturing the nuances of temporal dependencies. In this study, we use all of the mentioned methods including LR, KNN, SVM, DT, Bagging, RF, ET, AdaBoost, GB, and XGB to develop Voting approach.

3 Experiments

3.1 Data description

To assess the effectiveness of machine learning methods in imputing missing data, experiments are conducted using three real-world univariate time series. These datasets are selected from different fields, with different data collection frequencies. This ensures the generalizability of the imputation strategy. Details about each data set are presented in the table 2.

3.2 Experiments setting

To conduct experiments, we utilized Google Colaboratory for parallel training with 12GB RAM allocated per training session. We implemented ten machine learning algorithms from the Scikit-learn library for the imputation task: Linear Regression, K-Nearest Neighbors Regression, Support Vector Regression, Decision Tree and Extra Tree Regression, AdaBoost Regression, Bagging Regression, Gradient Boosting Regression, and Random Forest Regression), and Voting Regression. Additionally, we employed the XGBRegressor model from the XGBoost library to further enhance the imputation capabilities.

3.3 Experiment protocol

Indeed, we are unable to assess the performance of imputation algorithms on actual missing data. Consequently, to compare the effectiveness of imputation methods, we must generate simulated missing values on complete data. We employ a three-step method for evaluating the outcomes as follows: i) Step 1 creates simulated missing data. ii) and iii) the final step is to evaluate the performance of these methods.

– Creating missing data: In this study, for each dataset, different gaps with various sizes are created at random positions. Specifically, for the Phu Lien humidity dataset with monthly sampling, the gap size is 6, 12, 18, 24, and 36 data points respectively. It corresponds

Table 2: Datasets summary

Name of Data	Number Sample	Period
Phu Lien humidity	672 (Monthly)	1959 - 2014
Cua Ong temperature	9,858 (Daily)	1973 - 1999
Ha Noi water level	29,224 (3 hours)	2008 - 2017

to missing data for 6 months, 12 months, 18 months, 24 months, and 36 months. In the case of the Cua Ong temperature dataset, which features daily sampling, the missing gap sizes will range from 7, 14, 30, 90, and 180 data points. It corresponds to missing data for 1 week, 2 weeks, 1 month, 3 months, and 6 months. Meanwhile, for the Ha Noi water level dataset, with a sampling frequency of every 3 hours, the missing gap sizes to be addressed including 8, 16, 24, 40, and 56 data points. It corresponds to missing data for 1 day, 2 days, 3 days, 5 days, and 7 days.

– Filling missing data: the imputation algorithms are employed to fill in the missing values within the dataset.

– Evaluating the performance of ML methods: in this step, the performance of the imputation methods is assessed using different indicators as defined below.

3.4 Evaluation metric

To evaluate the performance of ML methods based on the proposal for the time series imputation task, we use the following metrics: Similarity (Sim), Normalized Mean Absolute Error (NMAE), Root Mean Square Error (RMSE), Fraction of Standard Deviation (FSD), Fractional Bias (FB) [25]. All metrics denote y as actual values and $\hat{y}$ as imputed values. we will split the actual part and then separate and remove it from the original data to simulate the missing scenario. For the forecasting data, the input of the model is considered as follows:

$$y = f(t_{T-n}, t_{T-n+1}, \dots t_T)$$

where $t_{T-n}, t_{T-n+1}, \dots t_T$ is the last n observation values in original data, T is the length of original data. After prediction, the result is concatenated into the previous input and removed from the first value of the set of inputs for the next forecasting value.

1. Similarity (Sim) - refers to the proportion that is similar between the real values (y) and the imputed value $\hat{y}$. It is computed as:

$$Sim(y, \hat{y}) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{1 + \frac{|y_i - \hat{y}_i|}{\max(\hat{y}) - \min(\hat{y})}}$$

where T is the number of missing values. A greater similarity (Sim value $\in$ $[0, 1]$) indicates a superior capacity to fill in the missing variables. In the case of constant ($\hat{y} =$ constant), we set $\max(\hat{y}) - \min(\hat{y}) = 1$.

2. Normalized Mean Absolute Error, NMAE is the average of the absolute difference between real values (y) and the imputed value $\hat{y}$:

$$NMAE(y, \hat{y}) = \frac{1}{T} \sum_{t=1}^{T} \frac{|y_i - \hat{y}_i|}{V_{max} - V_{min}}$$

Where V_{max} and V_{min} are the maximum and the minimum values of the input time series (time series has missing data) by ignoring the missing values. The range of NMAE value is from 0 to ∞. A better performance for the imputation task is indicated by a lower NMAE value.

3. Root Mean Square Error, RMSE is determined by calculating the mean of the squared differences between y and $\hat{y}$. This metric proves highly valuable in assessing overall precision or accuracy. Generally, a more effective method would exhibit a lower RMSE.

$$RMSE(y, \hat{y}) = \sqrt{\sum_{t=1}^{T} (y_i - \hat{y}_i)^2}$$

4. Fraction of Standard Deviation, FSD is determined by dividing the fraction of standard deviation (SD) between y and $\hat{y}$. This index indicates Whether a method is acceptable or not

$$FSD(y, \hat{y}) = 2 * \frac{|SD(y) - SD(\hat{y})|}{SD(y) + SD(\hat{y})}$$

For the imputation task, the imputation values are more relative to the actual values if FSD is closer to 0.

5. Fractional Bias, FB determines whether the imputation values are overestimated or underestimated relative to those observed.

$$FB(y, \hat{y}) = 2 * \frac{Mean(y) - Mean(\hat{y})}{Mean(y) + Mean(\hat{y})}$$

A model is considered perfect when its FB tends to 0 and acceptable when $-0.3 \leq FB \leq 0.3$.

4 Results and discussion

In this study, we focus on handling missing data in mono-dimensional time series data. Therefore, to be able to apply supervised ML methods, we first need to convert data from 1-dimensional to multi-dimensional data. We set up different schemes with the aim of finding out which dimension gives the best results.

- In the first scheme, the window size (i.e. the number of previous values used to predict the next one) is chosen exactly equal to the gap size. With the Phu Lien humidity dataset, the window sizes are 6, 12, 18, 24, and 36, respectively, equivalent to 6, 12, 18, 24, and 36 months. For Cua Ong temperature time series, the gap sizes are 7, 14, 30, 90, and 180 s equivalent to 1 week, 2 weeks, 1 month, 3 months, and 6 months. For the Hanoi water level data set, the gap sizes are 8, 24, 40, 56, and 112 corresponding to missing data for 1 day, 3 days, 5 days, 1 week, and 2 weeks.

- In the second scheme, we choose the window size to be equal to the missing data in 1 quarter. Specifically, for the data sets of Phu Lien humidity, Cua Ong temperature, and Hanoi water level, the number of previous data points used to predict the next value are 3, 90, and 720 points, respectively.

- In the third scheme, the window size is set to match the missing data in a year. In particular, the number of preceding data points utilized to forecast the next value for the datasets of Phu Lien humidity, Cua Ong temperature, and Hanoi water level is 12, 365, and 2920 data points, correspondingly.

For each gap, all the ML algorithms are conducted 10 times by randomly selecting the missing positions on the data. We then run 50 times for each data set and each window size.

Performance of WBDI.

Table 3, 4 and 5 demonstrate the average performance of eleven machine learning methods on three univariate time series datasets for the imputation task using various indices. The best results for each gap are highlighted in bold.

As depicted in Table 3, on the Phu Lien humidity time series with monthly frequency sampling, among the machine learning approaches, ensemble learning methods perform better than other ones. Especially, the AdaBoost and Bagging methods consistently indicate superior performance across all gap size configurations. These methods achieve the lowest NMAE, RMSE, and highest Similarity metrics on every missing rate. However, when considering FSD and FB indices, AdaBoost and Bagging do not achieve good results compared to quantitative indicators such as NMAE, RMSE, and Similarity. Following AdaBoost and Bagging is another ensemble learning ET method. This method gives the same performance as AdaBoost when the data is missing 1, 2, and 3 years, corresponding to 12, 24, and 36 missing points. Next, we can mention RF and GB methods. Although the two methods do not give good results at all missing levels like AdaBoost and Bagging, they do well at incomplete ratios of 12 months, and at other missing levels, it is always behind AdaBoost. KNN, LR, and SVM methods generate the least accurate results compared to ensemble learning methods for the Phu Lien humidity dataset.

For the Cua Ong temperature time series, data collected daily from 1973 to 1999, we see that the AdaBoost method once again demonstrates the ability to predict missing values in time series data (table 4). Considering quantitative metrics, AdaBoost always produces the highest Similarity results and the lowest NMAE and RMSE errors at all levels of missing data. Following AdaBoost is Bagging at missing data levels of 2 weeks, and 1 month, and this method gives the same performance as AdaBoost at missing rates of 1 week, 3 months, and 6 months, respectively, when considering quantitative indicators. Tree-based ensemble methods such as RF, ET, or GB are methods with performance right after the Bagging method. Especially for the Cuong temperature data set, the LR method does not show well the possibility of estimating missing data For the two indexes FSD and FB, the ML methods produce unstable results: sometimes it's RF, SVM, and sometimes AdaBoost.

Table 5 demonstrates the performance of eleven ML methods at five different gap sizes on the Hanoi water level dataset, with a sampling frequency of 3 hours. The results show that AdaBoost once again proves its ability to estimate incomplete data. This method gives good results of a 3/5 missing rate when paying attention to quantitative indicators. Specifically, This model holds the most elevated position in 3 gaps: 8, 56, and 80 missing points correspond to 1 day, 3 days, and 1 week. In this table, we can see that when predicting missing results within 1 day (8 consecutive missing points), all ML methods produce better results when estimating larger missing values. For the other gaps, the performance of machine learning methods varies, but AdaBoost still stands at the top 3 executed models regarding Similarity, NMAE, and RMSE indices. Following AdaBoost are other ensemble learning methods such as ET or Bagging. The result indicates, another time, the strength of the ensemble machine-learning method for imputating incomplete values in time series data. The results generated from LR and SVM are similar to the previous two data sets, Cua Ong temperature and Phu Lien humidity. These two methods indicate poorer performance compared to other methods when performing the predicting task of missing data.

In addition to comparing the performance of algorithms using evaluation metrics, in this study, we also conduct comparisons of how well different imputation techniques perform in terms of visualization. The prediction data generated by AdaBoost, RF, LR, and SVM algorithms on the Hanoi water level dataset is depicted in figure 7 with 5 days missing. We can see from this figure that AdaBoost accurately represents the shape and dynamics of real data. On

Table 3: Performance of different ML models on Phu Lien dataset for imputing missing data.

Gap Size	Model Name	Similarity	NMAE	RMSE	FSD	FB
6	DT	0.91	**0.10**	**0.11**	0.65	0.18
	ET	0.91	**0.10**	**0.11**	0.65	0.18
	AdaBoost	**0.92**	**0.10**	**0.11**	0.74	0.19
	GB	0.91	**0.10**	**0.11**	0.65	0.18
	XGB	0.87	0.15	0.18	0.45	0.16
	RF	0.91	0.11	0.13	0.45	0.23
	Bagging	**0.92**	**0.10**	**0.11**	0.74	0.19
	Voting	0.91	**0.10**	**0.11**	0.65	0.18
	LR	0.87	0.15	0.18	0.45	0.16
	KNN	0.91	0.11	0.13	0.45	0.23
	SVM	0.88	0.15	0.18	**0.29**	**0.09**
12	**DT**	**0.91**	**0.11**	**0.13**	0.58	-0.01
	ET	**0.91**	**0.11**	**0.13**	0.58	-0.01
	AdaBoost	**0.91**	**0.11**	**0.13**	0.58	-0.01
	GB	**0.91**	**0.11**	**0.13**	0.58	-0.01
	XGB	0.87	0.15	0.18	**0.44**	**0.00**
	RF	**0.91**	**0.11**	**0.13**	0.58	-0.01
	Bagging	**0.91**	**0.11**	**0.13**	0.58	-0.01
	Voting	**0.91**	**0.11**	**0.13**	0.58	-0.01
	LR	0.87	0.15	0.18	**0.44**	**0.00**
	KNN	**0.91**	**0.11**	**0.13**	0.58	-0.01
	SVM	0.87	0.15	0.18	**0.44**	**0.00**
18	DT	0.91	0.10	0.12	0.50	-0.01
	ET	0.91	0.10	0.13	0.57	**0.00**
	AdaBoost	**0.92**	**0.09**	**0.11**	0.40	0.08
	GB	0.91	0.10	0.13	0.57	**0.00**
	XGB	0.88	0.14	0.18	**0.24**	0.15
	RF	0.91	0.11	0.14	0.36	0.02
	Bagging	**0.92**	**0.09**	**0.11**	0.40	0.08
	Voting	0.91	0.11	0.14	0.36	0.02
	LR	0.91	0.11	0.14	0.36	0.02
	KNN	0.91	0.11	0.14	0.36	0.02
	SVM	0.91	0.11	0.14	0.36	0.02
24	**DT**	**0.92**	**0.09**	**0.12**	0.49	-0.05
	ET	**0.92**	**0.09**	**0.12**	0.49	-0.05
	AdaBoost	**0.92**	**0.09**	**0.12**	0.49	-0.05
	GB	**0.92**	**0.09**	**0.12**	0.49	-0.05
	XGB	0.87	0.16	0.21	**0.42**	0.06
	RF	0.91	0.11	0.15	0.52	**-0.04**
	Bagging	**0.92**	**0.09**	**0.12**	0.49	-0.05
	Voting	**0.92**	**0.09**	**0.12**	0.49	-0.05
	LR	0.87	0.16	0.21	**0.42**	0.06
	KNN	**0.92**	**0.09**	**0.12**	0.49	-0.05
	SVM	0.87	0.16	0.21	**0.42**	0.06
36	**DT**	**0.92**	**0.09**	**0.10**	**0.31**	**0.02**
	ET	**0.92**	**0.09**	**0.10**	**0.31**	**0.02**
	AdaBoost	**0.92**	**0.09**	**0.10**	**0.31**	**0.02**
	GB	0.89	0.13	0.16	0.57	0.10
	XGB	0.87	0.16	0.20	**0.31**	0.14
	RF	0.87	0.16	0.20	**0.31**	0.14
	Bagging	**0.92**	**0.09**	**0.10**	**0.31**	**0.02**
	Voting	0.89	0.13	0.16	0.57	0.10
	LR	0.87	0.16	0.20	**0.31**	0.14
	KNN	0.87	0.16	0.20	**0.31**	0.14
	SVM	0.87	0.16	0.20	**0.31**	0.14

the other hand, the other methods show significant discrepancies in capturing the characteristics of the data.

Performance of WBDI and some state-of-the-art methods.

We also conduct a separate analysis using the PhuLien dataset to evaluate the performance of WBDI (AdaBoost) and other state-of-the-art (SOTA) models across various gap sizes. The results, presented in Table 6, demonstrate promising outcomes, consistently outperforming traditional methods. The findings indicate that the WBDI (AdaBoost) model outperforms state-of-the-art methods (MLBUI_RF, MLBUI_SVM, and SAITS) across different gap sizes. WBDI consistently achieves higher similarity scores, reflecting a stronger alignment between predicted and actual data. As the gap size increases, WBDI exhibits decreasing NMAE and RMSE values, which suggest im-

Table 4: Performance of different ML models on Cua Ong dataset for imputing missing data.

Gap Size	Model Name	Similarity	NMAE	RMSE	FSD	FB
7	**DT**	**0.96**	**0.04**	**0.04**	1.11	-0.02
	ET	**0.96**	**0.04**	**0.04**	1.11	-0.02
	AdaBoost	**0.96**	**0.04**	**0.04**	1.11	-0.02
	GB	**0.96**	**0.04**	**0.04**	1.11	-0.02
	XGB	0.91	0.10	0.10	**0.52**	-0.12
	RF	0.92	0.09	0.11	0.89	**0.00**
	Bagging	**0.96**	**0.04**	**0.04**	1.11	-0.02
	Voting	0.92	0.09	0.11	0.89	**0.00**
	LR	0.91	0.10	0.10	**0.52**	-0.12
	KNN	0.92	0.09	0.11	0.89	**0.00**
	SVM	0.91	0.10	0.10	**0.52**	-0.12
14	DT	0.91	0.10	0.11	1.10	0.15
	ET	0.91	0.10	0.11	1.10	0.15
	AdaBoost	**0.92**	**0.09**	**0.10**	1.22	**0.09**
	GB	0.91	0.10	0.11	1.10	0.15
	XGB	0.90	0.12	0.14	**0.55**	0.16
	RF	0.90	0.12	0.14	**0.55**	0.16
	Bagging	0.91	0.10	0.11	1.10	0.15
	Voting	0.91	0.10	0.11	1.10	0.15
	LR	0.90	0.12	0.14	**0.55**	0.16
	KNN	0.91	0.10	0.11	1.10	0.15
	SVM	0.90	0.12	0.14	**0.55**	0.16
30	DT	0.91	0.11	0.14	0.84	0.18
	ET	0.91	0.11	0.14	0.84	0.18
	AdaBoost	**0.92**	**0.09**	**0.11**	1.33	0.06
	GB	0.91	0.11	0.14	0.84	0.18
	XGB	0.90	0.11	0.13	**0.79**	**0.04**
	RF	0.90	0.11	0.13	**0.79**	**0.04**
	Bagging	0.91	0.11	0.14	0.84	0.18
	Voting	0.90	0.11	0.13	**0.79**	**0.04**
	LR	0.90	0.11	0.13	**0.79**	**0.04**
	KNN	0.90	0.11	0.13	**0.79**	**0.04**
	SVM	0.90	0.11	0.13	**0.79**	**0.04**
90	DT	0.92	0.09	0.11	0.91	0.12
	ET	0.92	0.09	0.11	0.91	0.12
	AdaBoost	**0.93**	**0.07**	**0.09**	0.37	0.05
	GB	0.92	0.09	0.11	0.91	0.12
	XGB	0.87	0.17	0.20	0.54	**-0.03**
	RF	0.91	0.11	0.13	0.78	0.12
	Bagging	**0.93**	**0.07**	**0.09**	0.37	0.05
	Voting	0.92	0.09	0.11	0.35	-0.04
	LR	0.88	0.14	0.17	**0.34**	0.04
	KNN	0.91	0.11	0.13	0.78	0.12
	SVM	0.91	0.11	0.13	0.78	0.12
180	DT	0.98	0.02	**0.02**	1.07	0.02
	ET	0.98	0.02	**0.02**	1.07	0.02
	AdaBoost	**0.99**	**0.01**	**0.02**	0.83	0.02
	GB	0.98	0.02	**0.02**	1.07	0.02
	XGB	0.91	0.11	0.12	0.84	0.24
	RF	0.98	0.02	**0.02**	0.83	-0.02
	Bagging	**0.99**	**0.01**	**0.02**	0.83	0.02
	Voting	0.98	0.02	**0.02**	1.07	0.02
	LR	0.91	0.11	0.12	0.84	0.24
	KNN	0.98	0.02	**0.02**	0.85	**-0.01**
	SVM	0.98	0.02	0.03	**0.59**	0.01

proved accuracy in handling larger data gaps. This can be attributed to the robust nature of AdaBoost, which effectively combines multiple weak learners. In contrast, MLBUI_RF and MLBUI_SVM display more fluctuations in their performance. Although SAITS demonstrates competitive Similarity scores, its performance in terms of NMAE and RMSE generally lags behind WBDI. This suggests that while SAITS might capture certain patterns effectively, it may be less accurate in predicting numerical values. This further validates WBDI as a highly effective and robust approach for time series imputation with varying gap sizes.

Overall, the experimental results strongly emphasize WBDI (AdaBoost)'s consistent and superior performance across different gap sizes, highlighting its robustness and effectiveness in handling diverse data scenarios. These findings underscore the importance of leveraging ensemble

Table 5: Performance of different ML models on Ha Noi dataset for imputing missing data.

Gap Size	Model Name	Similarity	NMAE	RMSE	FSD	FB
	DT	0.98	0.02	**0.02**	1.07	0.02
	ET	0.98	0.02	**0.02**	1.07	0.02
	AdaBoost	**0.99**	**0.01**	**0.02**	0.83	0.02
	GB	0.98	0.02	**0.02**	1.07	0.02
	XGB	0.91	0.11	0.12	0.84	0.24
8	RF	0.98	0.02	**0.02**	0.83	-0.02
	Bagging	**0.99**	**0.01**	**0.02**	0.83	0.02
	Voting	0.98	0.02	**0.02**	1.07	0.02
	LR	0.91	0.11	0.12	0.84	0.24
	KNN	0.98	0.02	**0.02**	0.85	**-0.01**
	SVM	0.98	0.02	0.03	**0.59**	0.01
	DT	**0.97**	**0.03**	**0.03**	0.89	**-0.03**
	ET	**0.97**	0.04	0.04	0.52	**-0.03**
	AdaBoost	0.92	0.12	0.13	1.27	-0.18
	GB	0.90	0.14	0.14	0.63	-0.12
	XGB	0.90	0.14	0.14	0.83	-0.16
16	RF	0.91	0.13	0.14	0.96	-0.16
	Bagging	0.93	0.08	0.10	**0.33**	-0.16
	Voting	0.91	0.12	0.13	0.57	-0.16
	LR	**0.97**	0.04	0.04	0.83	-0.04
	KNN	0.96	0.04	0.05	0.82	0.04
	SVM	0.96	0.05	0.05	0.94	-0.09
	DT	**0.95**	0.06	0.07	1.17	0.10
	ET	**0.95**	0.06	0.07	1.17	0.10
	AdaBoost	**0.95**	**0.05**	**0.06**	0.65	0.09
	GB	**0.95**	0.06	0.07	1.17	0.10
	XGB	0.89	0.14	0.15	**0.55**	0.28
24	RF	0.94	0.07	0.07	0.65	0.11
	Bagging	0.93	0.09	0.10	0.72	**0.02**
	Voting	**0.95**	0.06	0.07	1.17	0.10
	LR	0.89	0.14	0.15	**0.55**	0.28
	KNN	0.94	0.07	0.07	0.65	0.11
	SVM	0.89	0.14	0.15	**0.55**	0.28
	DT	**0.97**	**0.04**	**0.05**	0.90	0.07
	ET	**0.97**	**0.04**	**0.05**	0.90	0.07
	AdaBoost	0.96	0.05	**0.05**	1.02	0.10
	GB	0.96	**0.04**	**0.05**	0.34	**-0.06**
	XGB	0.90	0.13	0.14	0.91	0.32
40	RF	0.96	**0.04**	**0.05**	0.34	**-0.06**
	Bagging	0.94	0.07	0.08	**0.20**	-0.22
	Voting	0.96	**0.04**	**0.05**	0.34	**-0.06**
	LR	0.90	0.13	0.14	0.91	0.32
	KNN	0.96	**0.04**	**0.05**	0.34	**-0.06**
	SVM	0.90	0.13	0.14	0.91	0.32
	DT	0.96	0.04	0.05	1.15	0.02
	ET	0.96	0.05	0.06	0.79	-0.03
	AdaBoost	**0.97**	**0.03**	**0.04**	0.84	-0.05
	GB	0.96	0.05	0.06	0.79	-0.03
	XGB	0.90	0.13	0.14	0.48	-0.23
56	RF	0.90	0.13	0.14	0.48	-0.23
	Bagging	**0.97**	**0.03**	**0.04**	0.84	-0.05
	Voting	0.95	0.05	0.07	**0.36**	**-0.01**
	LR	0.90	0.13	0.14	0.48	-0.23
	KNN	0.95	0.05	0.07	**0.36**	**-0.01**
	SVM	0.90	0.13	0.14	0.48	-0.23

techniques, particularly WBDI (AdaBoost), for achieving accurate and reliable predictions in imputation tasks.

5 Conclusion

This study introduces a novel framework called Weighted Bi-directional Imputation (WBDI) for handling missing data in univariate time series. WBDI leverages information from both before and after missing segments, incorporating weights to prioritize relevant information. This approach converts the data from univariate to multivariate format, allowing machine learning models to capture richer temporal dynamics. Experiments using eleven machine learning algorithms on three real-world datasets with varying sizes and sampling frequencies employed five evaluation metrics

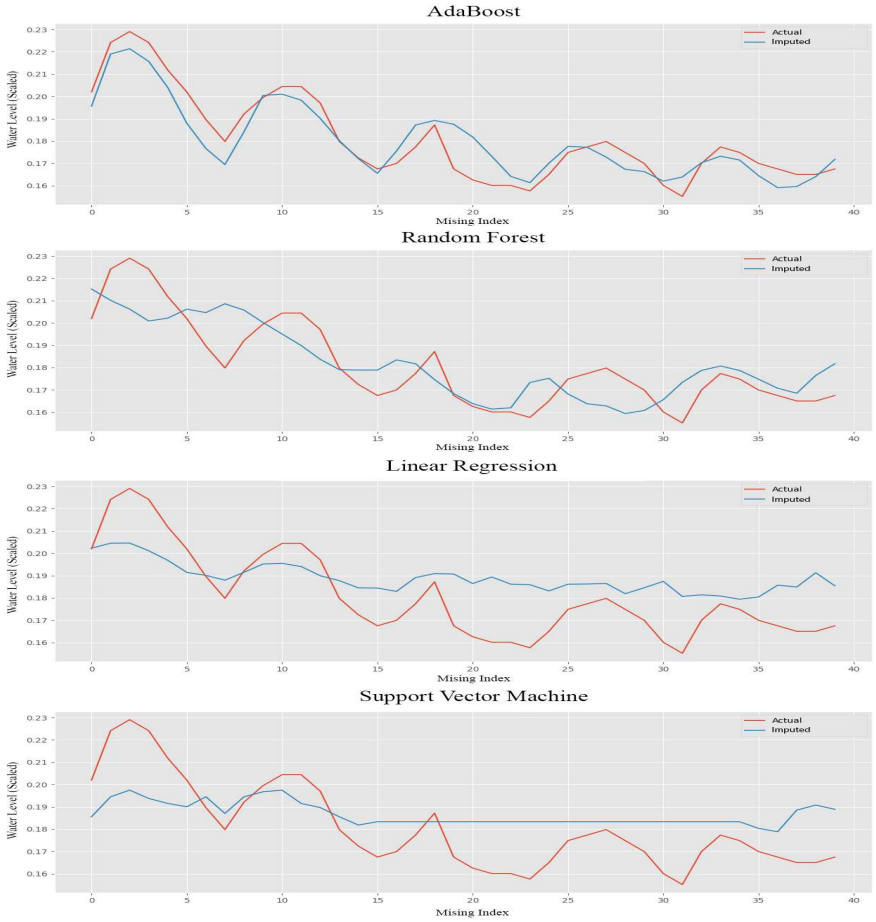

Figure 7: Comparison of true values with predicted ones generated from AdaBoost, RF, LR, and SVM methods on Ha Noi water level for 5 days missing data

Table 6: Comparing performance of WBDI (AdaBoost) with some SOTA methods on Phu Lien time series

Gap Size	Model Name	Similarity	NMAE	RMSE	FSD	FB
6	MLBUI_RF [25]	0.84	0.21	0.24	0.90	0.55
	MLBUI_SVM [25]	0.85	0.19	0.22	1.22	0.52
	SAITS [13]	0.90	0.12	0.16	2.00	0.37
	WBDI (AdaBoost)	**0.93**	**0.08**	**0.09**	**0.03**	**0.17**
12	MLBUI_RF [25]	0.91	0.11	0.14	1.03	0.12
	MLBUI_SVM [25]	0.89	0.13	0.15	0.96	0.11
	SAITS [13]	0.90	0.11	0.14	2.00	0.09
	WBDI (AdaBoost)	**0.92**	**0.09**	**0.10**	**0.45**	**0.04**
18	MLBUI_RF [25]	0.90	0.12	0.14	0.48	0.24
	MLBUI_SVM [25]	0.91	0.10	0.12	0.26	0.15
	SAITS [13]	0.88	0.14	0.18	2.00	0.26
	WBDI (AdaBoost)	**0.94**	**0.07**	**0.09**	**0.29**	**0.06**
24	MLBUI_RF [25]	0.89	0.13	0.15	0.77	0.02
	MLBUI_SVM [25]	0.90	0.12	0.15	0.91	0.01
	SAITS [13]	0.90	0.12	0.15	2.00	0.04
	WBDI (AdaBoost)	**0.93**	**0.08**	**0.10**	**0.43**	**0.06**
36	MLBUI_RF [25]	0.91	0.10	0.14	1.04	0.04
	MLBUI_SVM [25]	0.91	0.10	0.14	1.00	0.04
	SAITS [13]	0.90	0.12	0.14	2.00	0.12
	WBDI (AdaBoost)	**0.93**	**0.08**	**0.10**	**0.60**	**-0.02**

to assess performance. The results demonstrate the superiority of ensemble methods, particularly AdaBoost, which consistently achieved top performance across all configurations. These findings suggest that WBDI is a promising

approach for handling missing data in univariate time series. In the future, further exploration of deep learning techniques, such as recurrent neural networks, could potentially lead to even better imputation results.

References

[1] Suzanne Aigrain and Daniel Foreman-Mackey. Gaussian process regression for astronomical time series. *Annual Review of Astronomy and Astrophysics*, 61(1):329–371, 2023. https://doi.org/10.1146/annurev-astro-052920-103508.

[2] Paul D. Allison. Missing data. In *The Sage handbook of quantitative methods in psychology*, pages 72–89. Sage Publications Ltd, Thousand Oaks, CA, 2009. https://doi.org/10.4135/9780857020994.n4.

[3] Craig F. Ansley and Robert Kohn. On the estimation of ARIMA Models with Missing Values. In Emanuel Parzen, editor, *Time Series Analysis of Irregularly Observed Data*, Lecture Notes in Statistics, pages 9–37, New York, NY, 1984. Springer. https://doi.org/10.1007/978-1-4684-9403-7_2.

[4] Stefan Bauer, Bernhard Schölkopf, and Jonas Peters. The arrow of time in multivariate time series, 2016. https://doi.org/10.48550/arXiv.1603.00784.

[5] Leo Breiman. *Classification and Regression Trees*. Routledge, New York, October 1984. https://doi.org/10.1201/9781315139470.

[6] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, August 1996. https://doi.org/10.1007/BF00058655.

[7] Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, October 2001. https://doi.org/10.1023/A:1010933404324.

[8] Aizaz Chaudhry, Wei Li, Amir Basri, and François Patenaude. A Method for Improving Imputation and Prediction Accuracy of Highly Seasonal Univariate Data with Large Periods of Missingness. *Wireless Communications and Mobile Computing*, 2019:4039758, January 2019. https://doi.org/10.1155/2019/4039758.

[9] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):6085, 2018. https://doi.org/10.1038/s41598-018-24271-9.

[10] Tianqi Chen and Carlos Guestrin. XGBoost. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, aug 2016. https://doi.org/10.1145/2939672.2939785.

[11] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995. https://doi.org/10.1007/bf00994018.

[12] Thomas G. Dietterich. Ensemble Methods in Machine Learning. In *Multiple Classifier Systems*, Lecture Notes in Computer Science, pages 1–15, Berlin, Heidelberg, 2000. Springer. https://doi.org/10.1007/3-540-45014-9_1.

[13] Wenjie Du. A deep learning model to impute missing data in time series. Master's thesis, Concordia University, November 2021. Unpublished.

[14] Tlamelo Emmanuel, Thabiso Maupong, Dimane Mpoeleng, Thabo Semong, Banyatsang Mphago, and Oteng Tabona. A survey on missing data in machine learning. *Journal of Big Data*, 8(1):1–37, 2021. https://doi.org/10.1186/s40537-021-00516-9.

[15] Chenguang Fang and Chen Wang. Time series data imputation: A survey on deep learning approaches. *arXiv preprint arXiv:2011.11347*, 2020. https://doi.org/10.48550/arXiv.2011.11347.

[16] Yoav Freund and Robert E Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139, August 1997. https://doi.org/10.1006/jcss.1997.1504.

[17] Jerome H. Friedman. Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics*, 29(5):1189–1232, 2001. Publisher: Institute of Mathematical Statistics.

[18] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, April 2006. https://doi.org/10.1007/s10994-006-6226-1.

[19] Honggui Han, Meiting Sun, Huayun Han, Xiaolong Wu, and Junfei Qiao. Univariate imputation method for recovering missing data in wastewater treatment process. *Chinese Journal of Chemical Engineering*, 53:201–210, 2023. https://doi.org/10.1016/j.cjche.2022.01.033.

[20] Chisimkwuo John, Emmanuel J Ekpenyong, and Charles C Nworu. Imputation of missing values in economic and financial time series data using five principal component analysis approaches. *CBN Journal of Applied Statistics (JAS)*, 10(1):3, 2019. https://doi.org/10.33429/cjas.10119.3/6.

[21] Heikki Junninen, Harri Niska, Kari Tuppurainen, Juhani Ruuskanen, and Mikko Kolehmainen. Methods for imputation of missing values in air quality data sets. *Atmospheric Environment*, 38(18):2895–2907,

2004. https://doi.org/10.1016/j.atmosenv.2004.02.026.

[22] Maksims Kazijevs and Manar D. Samad. Deep imputation of missing values in time series health data: A review with benchmarking. *Journal of Biomedical Informatics*, 144:104440, 2023. https://doi.org/10.1016/j.jbi.2023.104440.

[23] Francisco Martínez, M. P. Frías, María Dolores Pérez, and A. J. Rivera. A methodology for applying k-nearest neighbor to time series forecasting. *Artificial Intelligence Review*, pages 1–19, 2019. https://doi.org/10.1007/s10462-017-9593-z.

[24] Steffen Moritz, Alexis Sardá, Thomas Bartz-Beielstein, Martin Zaefferer, and Jörg Stork. Comparison of different methods for univariate time series imputation in r. *ArXiv*, abs/1510.03924, 2015. /urlhttps://doi.org/10.48550/arXiv.1510.03924.

[25] T.-T.-H. Phan. Machine Learning for Univariate Time Series Imputation. In *2020 International Conference on Multimedia Analysis and Pattern Recognition (MAPR)*, pages 1–6, Ha Noi, Vietnam,, October 2020. IEEE. https://doi.org/10.1109/mapr49794.2020.9237768.

[26] Thi-Thu-Hong Phan, Emilie Poisson Caillault, and André Bigand. eDTWBI: Effective Imputation Method for Univariate Time Series. In Hoai An Le Thi, Hoai Minh Le, Tao Pham Dinh, and Ngoc Thanh Nguyen, editors, *Advanced Computational Methods for Knowledge Engineering*, Advances in Intelligent Systems and Computing, pages 121–132, Cham, 2020. Springer International Publishing. https://doi.org/10.1007/978-3-030-38364-0_11.

[27] Thi-Thu-Hong Phan, Emilie Poisson Caillault, Alain Lefebvre, and André Bigand. 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.

[28] Xiaogang Su, Xin Yan, and Chih-Ling Tsai. Linear regression. *WIREs Computational Statistics*, 4(3):275–294, 2012. https://doi.org/10.1002/wics.1198.

[29] Yukai Yang. modelling nonlinear vector economic time series. *Department of Economics and Business, Business and Social Sciences, Aarhus University, Aarhus, Denmark*, 28:29–30, 2012. https://doi.org/10.5772/intechopen.70825.