Long Short-Term Imputer: Handling Consecutive Missing Values in Time Series

Anonymous authors
Paper under double-blind review

Abstract

Encountered frequently in time series data, missing values can significantly impede time-series analysis. With the progression of deep learning, advanced imputation models delve into the temporal dependencies inherent in time series data, showcasing remarkable performance. This positions them as intuitive selections for time series imputation tasks which assume "Miss Completely at Random". Nonetheless, long-interval consecutive missing values may obstruct the model's ability to grasp long-term temporal dependencies, consequently hampering the efficacy of imputation performance. To tackle this challenge, we propose Long Short-term Imputer (LSTI) to impute consecutive missing values with different length of intervals. Long-term Imputer is designed using the idea of bi-directional autoregression. A forward prediction model and a backward prediction model are trained with a consistency regularization, which is designed to capture long-time dependency and can adapt to long-interval consecutive missing values. Short-term Imputer is designed to capture short-time dependency and can impute the short-interval consecutive missing values effectively. A meta-weighting network is then proposed to take advantage of the strengths of two imputers. As a result, LSTI can impute consecutive missing values with different intervals effectively. Experiments demonstrate that our approach, on average, reduces the error by 57.4% compared to state-of-the-art deep models across five datasets.

1 Introduction

Multivariate time series are abundant in real-world applications, including meteorology, finance, engineering, science, and healthcare (Hajihashemi & Popescu, 2015; Prakosa et al., 2012; Gupta et al., 2020; Tascikaraoglu et al., 2016; Zheng et al., 2015). These time series data often contain missing values due to reasons such as sensor failures and communication losses (Silva et al., 2012; Yi et al., 2016). These missing values impair the integrity and interpretability of the data, posing significant challenges to time series analysis (Hasan et al., 2021). Therefore, effectively imputing missing data before analysis is essential.

Many studies have addressed the task of imputing missing values using deep learning techniques (Wu et al., 2022; Du et al., 2023; Tashiro et al., 2021), and most of them are designed for scenarios involving Missing Completely at Random (MCAR). However, in the real world, missing values often occur consecutively and in clusters due to factors such as signal loss, environmental interference, and equipment failure. For example, when a car travels through a tunnel, all satellite signals related to the car will be blocked by the tunnel, resulting in a long interval of consecutive missing values in the time series data.

Following the scheme in Khayati et al. (2020), we categorize the consecutive time series missing patterns into four distinct types: 'Disjoint', 'Overlap', 'MCAR_B', and 'Blackout'. Each pattern represents a specific form of long-interval consecutive missing data. This paper concentrates on the 'Blackout' pattern, considered the most challenging, where missing values across all channels are aligned in the same positions. This synchronized absence of data introduces significant difficulties for standard imputation methods, as it eliminates the possibility of leveraging information from other channels or time points to reconstruct the missing values. Figure 1 illustrates the differences between MCAR and Blackout missing patterns.

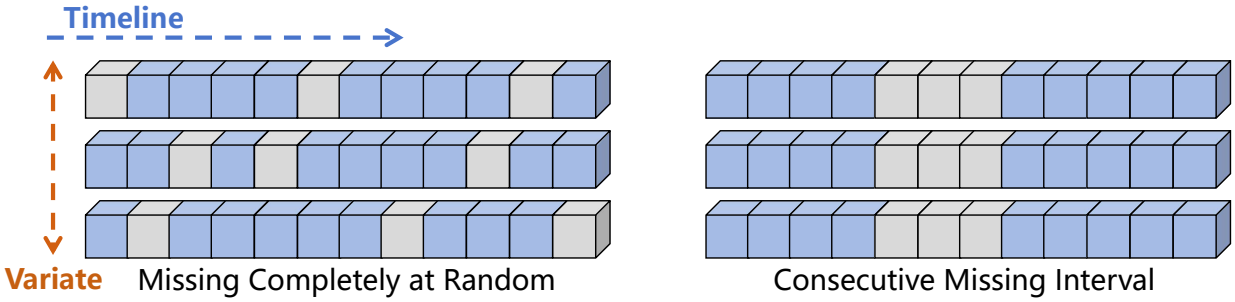


Figure 1: The difference between MCAR and long-interval consecutive missing values.

Traditional imputation methods perform poorly on this type of data due to their inadequate long-term modeling capabilities. Some methods attempt to address this problem in specific application domains (Ma et al., 2020; Chen et al., 2021; Guo et al., 2021), and other methods utilize mathematical and statistical techniques to tackle this issue (Wongoutong, 2020). However, these methods require domain knowledge and complex processing workflows, and they are not end-to-end deep learning solutions, thus limiting the range of problems they can effectively solve.

To address the problem of imputing Blackout data, we propose LSTI, a Long Short-Term Imputer specifically designed for this purpose. The model consists of three main components: first, a Long-Term Imputer, which is composed of a forward prediction network and a backward prediction network. During training, an additional consistency loss is introduced to minimize the discrepancy between the prediction of two networks. During imputation, the two prediction networks autoregressively impute the entire sequence in forward and backward directions, respectively, with the final imputation result obtained as a weighted sum of the two which follows the idea that the closer the values in the prediction window are to the lookback window, the smaller cumulative errors are. This structure effectively captures long-term dependencies in time series data and is suitable for imputing long-interval consecutive missing values.

The second component is the Short-Term Imputer. Missing data in real-world is often a mixture of various missing patterns, including MCAR and long-interval consecutive missing values. This requires a model that can adaptively impute different types of missing data. The Short-Term Imputer consists of a self-mapping network, trained with a randomly generated consecutive missing mask to capture short-term dependencies in the time series data. Subsequently, LSTI employs a Meta-weighting module to learn the importance of long-term and short-term dependencies in the current data, and uses this to weight the outputs of the long-term and short-term imputers to obtain the final imputation result. This structure can simultaneously learn both long-term and short-term dependencies in time series data, enabling adaptive and effective imputation across various missing data patterns.

Our main contributions are summarized as follows:

- We propose the Long-Term Imputer to capture long-term dependencies in time series data. It uses forward and backward prediction networks to autoregressively impute missing values from both directions. Additionally, a consistency loss is introduced during training to minimize the discrepancy between the two predictions.

- We propose the Short-Term Imputer to capture short-term dependencies in time series data. A Meta-weighting module is used to learn the proportion of long-term and short-term dependencies in specific data, adaptively balancing the weights of long-term and short-term dependencies. This allows LSTI to effectively impute data under various missing patterns.

- We conduct extensive experiments on five real-world datasets, and our method outperformed the current state-of-the-art imputation methods on all datasets.

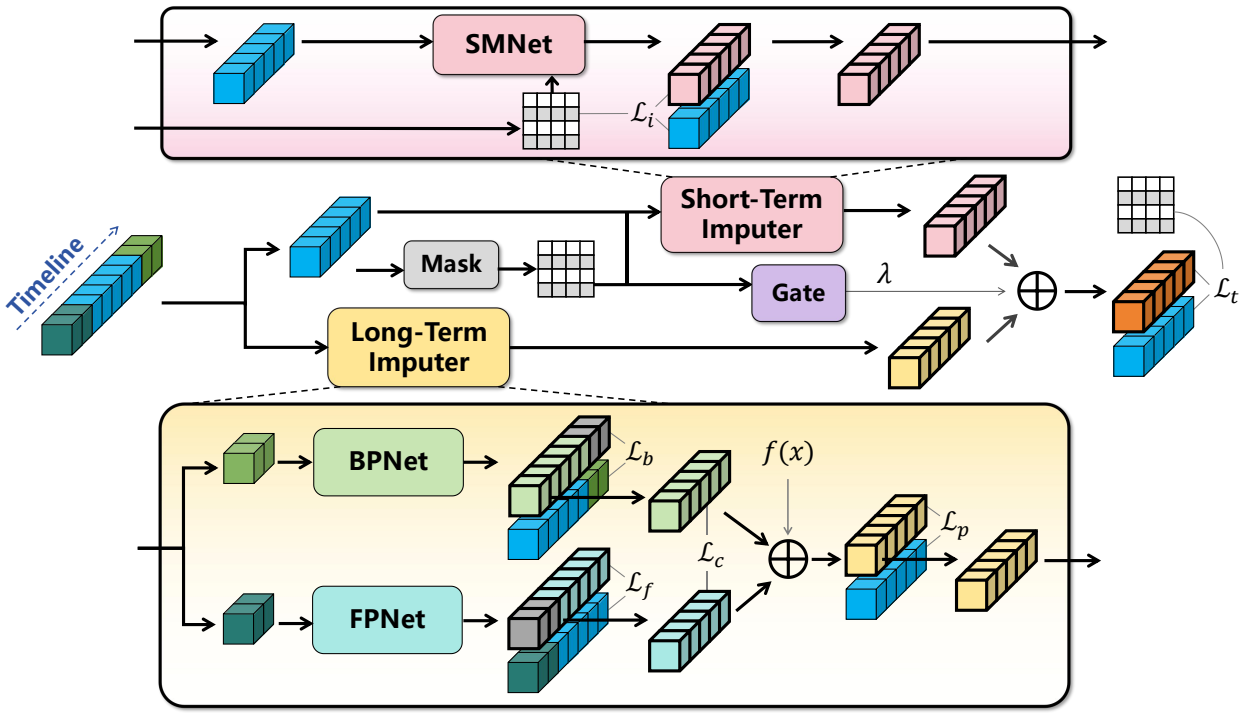


Figure 2: The overall training process of LSTI. LSTI consists of three main parts: (1) the Long-Term Imputer, which is composed of two prediction networks operating in different directions; (2) the Short-Term Imputer, which consists of a self-mapping module; (3) the Meta-weighting module, which learns to weigh the outputs of the two Imputers and can adapt to specific input data.

2 Related Work

2.1 Time Series Imputation

Deep learning models are inherently good at capturing complex and non-linear relationships in data. RNN-based models are more commonly used in earlier works. GRU-D (Che et al., 2018), a variant of the Gated Recurrent Unit (GRU), addresses missing data in time series classification tasks. Soon after, BRITS (Cao et al., 2018) imputes missing values using a bidirectional recurrent dynamical system, without specific assumptions. In the case of M-RNN (Yoon et al., 2018), it imputes missing values based on hidden states derived from bidirectional RNNs. Since the generation ability is naturally suited to the imputation task, generative models are then widely used in filling missing values. E2GAN (Luo et al., 2019) incorporates a generator based on GRUI within an auto-encoder framework. For spatiotemporal sequence imputation, NAOMI (Liu et al., 2019) introduces a non-autoregressive model to comprise a bidirectional encoder and a multiresolution decoder. With diffusion models gaining popularity, CSDI (Tashiro et al., 2021) emerges, which is a conditional score-based diffusion model designed for time-series imputation. Advancing further, TimesNet (Wu et al., 2022) expands the examination of temporal variations into the two-dimensional space, while considering the presence of multiple periodicity in time series data. NRTSI (Shan et al., 2023) is an approach for time-series imputation that treats time series as a collection of (time, data) pairs. SAITS (Du et al., 2023) conducts simultaneous reconstruction and imputation by employing a weighted combination of two diagonally-masked self-attention blocks.

Each method discussed above possesses distinct advantages, but they are all designed for scenarios involving Missing Completely at Random (MCAR). When handling long-interval consecutive missing data, these methods may produce suboptimal results. Some methods attempt to address the imputation of consecutive missing data using specific domain knowledge in specific applications, such as air pollution (Ma et al., 2020), water quality prediction (Chen et al., 2021), and electrical engineering (Guo et al., 2021). Another method

attempts to tackle this problem using a statistical approach (Wongoutong, 2020). However, these approaches require complex processing workflows and are not end-to-end methods, leading to limited applicability.

2.2 Time Series Forecasting

Time series forecasting methods potentially can be used to predict long-interval consecutive missing data. Specifically, the values prior to the missing interval are used to predict the entire missing segment. Some RNN-based models demonstrate robust forecasting performance in earlier studies, such as LSTNet (Lai et al., 2018), which combines networks of DNNs, RNNs, and Skip-RNNs. DeepAR (Salinas et al., 2020) utilizes an autoregressive recurrent network architecture to output the probabilities of prediction points. In more recent research, Transformer-based models gain significant prominence. Early on, Informer (Zhou et al., 2021) utilizes self-attention distilling to significantly accelerate the training efficiency of Transformers; following this, Autoformer (Wu et al., 2021) embeds time series decomposition into the model, enhancing its capability to discover temporal dependencies; Non-stationary Transformers (Liu et al., 2022) takes a novel approach by incorporating non-stationarity factors into the attention mechanism to prevent the over-stabilization of time series data; PatchTST (Nie et al., 2022) achieves remarkable results through the use of patches and channel independence; most recently, iTransformer (Liu et al., 2023) transposes the input matrix and reverses the roles of the attention mechanism and feed-forward network, better capturing the correlations among multivariate time series.

Although forecasting methods appear promising for addressing long-interval consecutive missing data, various issues arise in practice. First, a series requiring imputation may have multiple missing points, and there is no guarantee that each missing segment is preceded by a complete, available segment for forecasting. Second, the endpoints of the predicted sequence cannot be effectively aligned with the observed values following the missing segment, leading to deviations in later forecasted values from actual values. Therefore, forecasting methods cannot be directly applied to impute long-interval consecutive missing data.

3 Long Short-Term Imputer

Problem Formulation We consider a collection of multivariate time series $\mathbf{X} = \{x_{1:T,1:C}\} \in \mathbb{R}^{T \times C}$ with T timestamps and C channels (attributes). The imputation task is to impute the missing values in $\mathbf{X}$. Formally, an observation mask is defined as $\mathbf{M} = \{m_{1:T,1:C}\} \in \mathbb{R}^{T \times C}$ where $m_{t,c} = 0$ if $x_{t,c}$ is missing, and $m_{t,c} = 1$ if $x_{t,c}$ is observed.

In practical scenarios, consecutive missing values can impact the matrix $\mathbf{M}$. The initial length of consecutive missing values is denoted as L_m. There are random submatrices $\mathbf{X}_{a_k:a_k+L_m-1,1:C} \subset \mathbf{X}$ where values are missing. We define $\mathbf{Q}_{a_k}$ as the observation mask which only considers the missing values of $\mathbf{X}_{a_k:a_k+L_m-1,1:C}$ in $\mathbf{X}$. Specifically, we have:

$$\mathbf{M} = \prod_{k=1}^{K} \mathbf{Q}_{a_k}, \quad (\mathbf{Q}_{a_k})_{ij} = \begin{cases} 0, & \text{if } a_k \leq i < a_k + L_m \\ 1, & \text{otherwise} \end{cases}, \tag{1}$$

where a_k represents the randomly selected starting position of the missing interval, and K denotes the total number of missing intervals that appeared in the original data.

Motivation Current imputation methods generally assume that test data is missing completely at random (MCAR). However, in the real world, missing values often occur consecutively and in clusters due to factors such as signal loss, environmental interference, and equipment failure. Mainstream imputation methods may not perform well when dealing with long-interval consecutive missing data because they are limited at learning long-term dependencies. Based on this practical concern, we design the Long Short-Term Imputer (LSTI), which can effectively impute consecutively missing data by learning both long-term dependencies and short-term dependencies with 2 specially designed expert models.

3.1 Overall structure

The LSTI model consists of the following modules: (1) Long-term Imputer: This module captures long-term dependencies in time series data. It includes forward and backward prediction networks, which use an additional consistency loss during training to minimize the discrepancy between them. During imputation, the forward and backward networks autoregressively impute the sequence in both directions, followed by a weighted ensemble for each missing interval. (2) Short-term Imputer: This module captures short-term dependencies in time series data. It includes a self-mapping imputation network, trained with a randomly generated consecutive missing mask. During imputation, it uses a sliding window approach for local imputation of the sequence. (3) Meta-weighting module: This module learns to weigh the outputs of the long-term and short-term Imputer considering the properties of specific input data. The final imputation result is obtained by ensembling the outputs of two imputers using the Meta-weighting approach. The overall framework of LSTI is shown in Figure 2. It is worth mentioning that the backbone of Long-term Imputer and Short-term Imputer can be any advanced backbone designed for time series, e.g., transformer-based model, RNN-based model or TCN-based model.

3.2 Long-term Imputer

The long-interval consecutive missing values may hinder the model from learning long dependencies. To alleviate the problem, two insights are employed to model long-term dependencies. First, the long-term dependencies can be learned in two directions. Thus Long-term Imputer includes a forward prediction network and a backward prediction network, which aim to learn long-term dependencies in each direction and impute missing data in an autoregressive manner. Second, the prediction results using the aforementioned two kinds of long-term dependencies should be consistent. As a result, during training, we incorporate a consistency loss to reduce the output discrepancy between the two networks to further enhance the effectiveness of learning long-term dependencies.

3.2.1 Training process

Bidirectional Prediction. Denote the input length of the prediction network as S and the prediction length as L, for a training dataset $\mathbf{X}_{tr} \in \mathbb{R}^{(S+L+S)\times C}$ composed of input for forward prediction $\mathbf{X}_f \in \mathbb{R}^{S\times C}$, the ground-truth values to be predicted $\mathbf{X}_t \in \mathbb{R}^{L\times C}$, input for backward prediction $\mathbf{X}_b \in \mathbb{R}^{S\times C}$, we reverse time order of $\mathbf{X}_b$ to obtain $\mathbf{X}'_b$. Specifically, we have:

$$\mathbf{X}_{tr} = [\mathbf{X}_f, \mathbf{X}_t, \mathbf{X}_b], \quad (x'_b)_{i,j} = (x_b)_{S-i,j}. \tag{2}$$

This indicates that $\mathbf{X}_f$ is a submatrix consisting of the first S rows of $\mathbf{X}_{tr}$, while $\mathbf{X}'_b$ is a submatrix formed by reversing the last S rows of $\mathbf{X}_{tr}$. $\mathbf{X}_f$ and $\mathbf{X}'_b$ are fed into the forward prediction network (FPNet) and the backward prediction network (BPNet), respectively, to obtain the network outputs $\mathbf{Y}_f, \mathbf{Y}'_b \in \mathbb{R}^{(S+L)\times C}$. $\mathbf{Y}_f$ consists of a predicted lookback window $\mathbf{Y}_{fl} \in \mathbb{R}^{S\times C}$ and a prediction window $\mathbf{Y}_{fp} \in \mathbb{R}^{L\times C}$, and similarly for $\mathbf{Y}'_b$. Specifically, we have:

$$\begin{aligned} \mathbf{Y}_f &= \mathrm{FPNet}(\mathbf{X}_f), \quad \mathbf{Y}'_b = \mathrm{BPNet}(\mathbf{X}'_b), \\ \mathbf{Y}_f &= [\mathbf{Y}_{fl}, \mathbf{Y}_{fp}], \qquad \mathbf{Y}'_b = [\mathbf{Y}'_{bp}, \mathbf{Y}'_{bl}], \end{aligned} \tag{3}$$

where $\mathbf{Y}'_b$, $\mathbf{Y}'_{bl}$, $\mathbf{Y}'_{bp}$ is the reversal of $\mathbf{Y}_b$, $\mathbf{Y}_{bl}$, $\mathbf{Y}_{bp}$. The network outputs both the lookback window and the prediction window values to ensure that the model's output matches its input, resulting in a comprehensive training process and a more stable autoregressive process. Therefore, the forward prediction loss $\mathcal{L}_f$ and backward prediction loss $\mathcal{L}_b$ need to match both the lookback window and the prediction window values:

$$\mathcal{L}_f = \mathrm{MSE}([\mathbf{X}_f, \mathbf{X}_t], \mathbf{Y}_f), \quad \mathcal{L}_b = \mathrm{MSE}([\mathbf{X}_t, \mathbf{X}_b], \mathbf{Y}_b). \tag{4}$$

Consistency Loss. The prediction windows output $\mathbf{Y}_{fp}, \mathbf{Y}_{bp}$ by the two networks corresponding to the same segment, $\mathbf{X}_t$. If the long-term dependencies can be captured by two networks, their predicted values for the same segment should be as close as possible. As a result, we propose consistency loss to further reduce potential error accumulation in the autoregressive process:

$$\mathcal{L}_c = \mathrm{MSE}(\mathbf{Y}_{fp}, \mathbf{Y}_{bp}). \tag{5}$$

Emsembling. Given predictions from both directions, we can ensemble them as the output of Long-term Imputer. An intuitive idea is that the closer the values in the prediction window are to the lookback window, the smaller cumulative errors are. Therefore, values at the beginning of the interval should be more aligned with the forward network's output, while values at the end of the interval should be more aligned with the backward network's output. We use a linear function $f(i)$ to weigh and ensemble the two prediction outputs, and compute the loss between the final sequence and the ground truth:

$$f(i) = \frac{i-1}{L-1}, \quad \mathbf{Y}_t = (1-f) \cdot \mathbf{Y}_{lp} + f \cdot \mathbf{Y}_{bp}, \quad \mathcal{L}_p = \mathrm{MSE}(\mathbf{X}_t, \mathbf{Y}_t). \tag{6}$$

The overall training objective of Long-term Imputer $\mathcal{L}_l$ is the sum of the aforementioned losses:

$$\mathcal{L}_l = \mathcal{L}_f + \mathcal{L}_b + \mathcal{L}_c + \mathcal{L}_p. \tag{7}$$

3.2.2 Imputation process

In real-world datasets, there may be multiple consecutive missing segments, which can result in missing values in the network's input. To address this issue, the Long-term Imputer uses a bidirectional autoregressive approach to impute the entire missing data at once during the imputation process. This ensures that the data input to the network is either a true value or a previously imputed value, thereby avoiding the issue of missing data in the input.

Bidirectional Autoregression. Consider the case of the forward prediction network. We scan the entire dataset from front to back along the time dimension. If missing values are found in the data segment $\mathbf{D}_k \in \mathbb{R}^{L \times C}$, and the corresponding missing mask is $\mathbf{M}_k$, we select the preceding data segment $\mathbf{D}_{kl} \in \mathbb{R}^{S \times C}$ from $\mathbf{D}_k$ and input it into the forward prediction network:

$$[\mathbf{E}_{kl}, \mathbf{E}_{kp}] = \mathrm{FPNet}(\mathbf{D}_{kl}), \quad \hat{\mathbf{D}}_k = \mathbf{M}_k \cdot \mathbf{D}_k + (1 - \mathbf{M}_k) \cdot \mathbf{E}_{kp}, \tag{8}$$

where $\hat{\mathbf{D}}_k$ is the imputed $\mathbf{D}_k$ sequence. We use $\hat{\mathbf{D}}_k$ to replace $\mathbf{D}_k$, ensuring that this sequence no longer contains any missing values. Then, we continue scanning the entire dataset and process the next segment $\mathbf{D}_{k+1}$.

The backward prediction network performs the same autoregressive imputation for the entire missing data from the opposite direction. Subsequently, we rescan the entire original missing data, and for each consecutive missing interval $\mathbf{H}_k \in \mathbb{R}^{L_k \times C}$, we emsemble the forward and backward imputation results using the method in Equation 6 to obtain the final imputed sequence $\hat{\mathbf{H}}$.

3.3 Short-term Imputer

Missing data in the real world is typically represented as a combination of single-point missing values and long consecutive missing intervals. To accommodate various missing patterns, we follow the idea of vanilla imputation models and introduce a Short-term Imputer to capture short-term dependencies in time series data. This module consists of a single self-mapping network (SMNet). For the training data in Short-term Imputer, $\mathbf{X}_t \in \mathbb{R}^{L \times C}$, which is the middle segments of training data in Long-term Imputer (which is also illustrated in Figure 2), is used as input, and we randomly generate a consecutive missing mask $\mathbf{M}_t$ and input them into the self-mapping network:

$$\mathbf{Z}_t = \mathrm{SMNet}(\mathbf{X}_t, \mathbf{M}_t), \quad \mathcal{L}_s = \mathrm{MSE}((1 - \mathbf{M}_t) \cdot \mathbf{X}_t, (1 - \mathbf{M}_t) \cdot \mathbf{Z}_t). \tag{9}$$

During imputation, we partition the entire dataset using a sliding window of length L and input the segments into the Short-term Imputer for imputation. Finally, the Short-term Imputer outputs the imputed data $\hat{\mathbf{I}}$.

3.4 Meta-weighting module

To adaptively impute data with a mixture of long-term and short-term missing values, we designed the Meta-weighting module. This module contains a standalone MLP that learns the characteristics of the missing

data to adaptively allocate weights to the long-term and short-term imputer outputs. During training, after obtaining the output $\mathbf{Y}_t$ from the Long-term Imputer and $\mathbf{Z}_t$ from the Short-term Imputer, we input the data $\mathbf{X}_t$ and the missing mask $\mathbf{M}_t$ into the Meta-weighting module:

$$\lambda = \text{MLP}(\mathbf{X}_t, \mathbf{M}_t), \quad \hat{\mathbf{X}}_t = (1-\lambda) \cdot \mathbf{Y}_t + \lambda \cdot \mathbf{Z}_t. \tag{10}$$

The training loss for Meta-weighting output is:

$$\mathcal{L}_t = \text{MSE}((1-\mathbf{M}_t) \cdot \mathbf{X}_t, (1-\mathbf{M}_t) \cdot \hat{\mathbf{X}}_t). \tag{11}$$

The overall training objective of LSTI is the sum of the losses from all modules:

$$\mathcal{L} = \mathcal{L}_l + \mathcal{L}_s + \mathcal{L}_t. \tag{12}$$

During imputation, after obtaining the output $\hat{\mathbf{H}}$ from the Long-term Imputer and $\hat{\mathbf{I}}$ from the Short-term Imputer, we use the same sliding window as the Short-term Imputer to get $\mathbf{D}_k$ and its missing mask $\mathbf{M}_k$ from the original data. These partitions are then input into the Meta-weighting module to obtain λ_k. For each window, we use the method in Equation 10 to meta-weighting $\hat{\mathbf{H}}_k$ and $\hat{\mathbf{I}}_k$ in proportion to get the final imputed data.

4 Experiments

In this section, we conduct experiments on five real-world datasets and compare them with the current mainstream imputation and forecasting methods. The experimental results indicate that our approach significantly outperforms existing methods in cases of long-interval consecutive missing data.

4.1 Experimental Setup

Datasets We use five real-world datasets in experiments. (1) **Electricity** (UCI) collects hourly electricity consumption data of 321 customers from 2012 to 2014. (2) **Traffic** (PeMS) contains hourly road occupancy rates measured by 862 sensors on San Francisco Bay area freeways from January 2015 to December 2016. (3) **METR-LA** (Metro) records four months of statistics on traffic speed on 207 sensors on the highways of Los Angeles County. (4) **Guangzhou** (Chen et al., 2018) records traffic speeds per ten minutes on 214 anonymous roads in Guangzhou from August 1, 2016 to September 30, 2016. (5) **PEMS04** (Chen et al., 2001) is a subset of PEMS, collected by 307 detectors over a continuous period of 59 days, starting from January 1, 2018.

Baselines We compare our method with several current mainstream imputation approaches, including **Transformer** (Vaswani et al., 2017), **SAITS** (Du et al., 2023), **TimesNet** (Wu et al., 2022), **CSDI** (Tashiro et al., 2021), and **ImputeFormer** (Nie et al., 2024). Additionally, we also conduct comparisons with long-term forecasting methods using **TimesNet** (Wu et al., 2022) and **iTransformer** (Liu et al., 2023) to validate the effectiveness of our approach.

Generating missing data To simulate the scenario of consecutive missing data in the real world, we generate missing datasets using the following method: For a given missing length L_m and a missing rate R_m, we randomly select intervals of length L_m in the data and mark them as missing until the overall missing rate R_m is reached. In our experiments, we conduct controlled variable comparisons by keeping either L_m or R_m constant while varying the other.

Implementation details Our LSTI is a general training framework. Hence the backbone of FPNet, BPNet and SMNet can be any advanced deep network specially designed for time series. In our experiments, we use TimesNet as the backbone in LSTI, employing the same hyperparameter settings as in the original paper, namely S is 96 and L is 96. In other baseline models, hyperparameters are set according to the configurations specified in their original papers. We use MAE and MSE as the metrics for our experiments.

Table 1: Imputation results on real datasets with different missing lengths and a fixed 30% missing rate. The label "_F" indicates models using forecasting methods.

		LSTI		Transformer		CSDI		SAITS		TimesNet		Imputeformer		TimesNet_F		iTransformer_F	
	R_m	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
Electricity	10	**0.249**	**0.138**	0.739	0.842	0.441	0.424	0.785	0.936	0.494	0.474	0.759	0.868	0.348	0.245	0.382	0.297
	30	**0.242**	**0.129**	0.782	0.923	0.546	0.616	0.798	0.965	0.571	0.618	0.788	0.926	0.330	0.222	0.372	0.290
	50	**0.255**	**0.144**	0.806	0.961	0.658	0.765	0.809	0.982	0.686	0.844	0.793	0.920	0.375	0.296	0.395	0.339
	100	**0.278**	**0.171**	0.830	1.029	0.774	1.005	0.828	1.041	0.784	1.008	0.814	0.976	0.433	0.407	0.434	0.414
	300	**0.370**	**0.251**	0.858	1.088	0.837	1.107	0.835	1.063	0.820	1.079	0.815	0.976	0.699	0.805	0.695	0.802
Traffic	10	**0.269**	**0.296**	0.743	1.129	0.590	0.816	0.762	1.158	0.603	0.830	0.739	1.143	0.374	0.456	0.401	0.497
	30	**0.281**	**0.316**	0.750	1.134	0.599	0.863	0.762	1.150	0.654	0.951	0.742	1.132	0.402	0.511	0.408	0.504
	50	**0.301**	**0.341**	0.762	1.164	0.555	0.871	0.769	1.166	0.697	1.056	0.750	1.124	0.389	0.478	0.373	0.454
	100	**0.334**	**0.381**	0.773	1.194	0.546	0.808	0.777	1.195	0.790	1.298	0.771	1.193	0.470	0.614	0.476	0.630
	300	**0.450**	**0.547**	0.782	1.213	0.635	0.972	0.782	1.210	0.785	1.240	0.769	1.207	0.663	0.986	0.670	1.002
METR-LA	10	**0.342**	**0.353**	0.674	1.119	0.411	0.826	0.580	0.881	0.460	0.636	0.677	1.006	0.565	0.848	0.644	1.074
	30	**0.434**	**0.568**	0.756	1.203	0.543	1.129	0.730	1.340	0.559	0.862	0.804	1.368	0.586	0.933	0.625	1.060
	50	**0.508**	**0.747**	0.795	1.272	0.586	1.345	0.742	1.381	0.559	0.926	0.773	1.216	0.592	0.951	0.612	0.987
	100	**0.478**	**0.714**	0.652	0.831	0.625	1.527	0.710	1.363	0.617	1.083	0.720	1.223	0.611	1.008	0.610	0.966
	300	**0.712**	**1.225**	0.842	1.567	0.768	1.729	0.811	1.658	0.812	1.476	0.773	1.365	0.817	1.519	0.828	1.527
Guangzhou	10	**0.367**	**0.293**	0.689	0.823	0.487	0.469	0.758	1.001	0.659	0.773	0.709	0.832	0.447	0.396	0.508	0.513
	30	**0.387**	**0.323**	0.698	0.845	0.606	0.678	0.747	0.975	0.797	1.080	0.714	0.871	0.449	0.411	0.512	0.528
	50	**0.395**	**0.321**	0.750	0.920	0.581	0.664	0.752	0.971	0.893	1.293	0.787	0.930	0.527	0.533	0.588	0.633
	100	**0.420**	**0.350**	0.748	0.909	0.588	0.672	0.752	0.938	0.828	1.126	0.762	0.915	0.540	0.545	0.559	0.584
	300	**0.495**	**0.466**	0.758	0.968	0.635	0.749	0.794	1.084	0.943	1.411	0.758	0.986	0.670	0.790	0.669	0.794
PEMS04	10	**0.213**	**0.099**	0.662	0.622	0.294	0.160	0.714	0.741	0.461	0.389	0.820	0.908	0.485	0.495	0.649	0.808
	30	**0.263**	**0.132**	0.779	0.833	0.457	0.356	0.760	0.820	0.685	0.828	0.813	0.908	0.523	0.668	0.648	0.901
	50	**0.267**	**0.142**	0.827	0.897	0.553	0.505	0.762	0.813	0.711	0.865	0.854	0.965	0.531	0.611	0.531	0.532
	100	**0.425**	**0.328**	0.888	1.037	0.706	0.767	0.867	1.020	0.872	1.126	0.876	1.015	0.612	0.680	0.627	0.658
	300	**0.569**	**0.513**	0.883	1.025	0.731	0.785	0.899	1.097	0.891	1.167	0.923	1.096	0.776	0.876	0.800	0.906

4.2 Main Results

To investigate the imputation capabilities of LSTI in various scenarios and its performance under different missing lengths L_m and missing rates R_m, we conduct controlled variable comparisons by keeping either L_m or R_m constant while varying the other. Experimental results show that our method outperforms the current state-of-the-art models in all cases.

4.2.1 Varying missing length

We investigate the imputation performance of different methods on real datasets with varying missing lengths, and the corresponding results are presented in Table 1. From the table, it can be observed that our method outperforms traditional imputation and forecasting methods across all datasets. For instance, regarding the MSE metric, our method leads to an average improvement of 72.32%, 55.43%, 40.27%, 54.54%, 68.29% compare to other methods on Electricity, Traffic, METR-LA, Guangzhou, and PEMS04 datasets, respectively. Additionally, we observe that as the length of consecutive missing data increases, the performance of traditional imputation and forecasting methods declines significantly, whereas our method exhibits a smaller decline across most datasets. This indicates that the length of consecutive missing data has a substantial impact on traditional imputation methods, whereas our method effectively imputes consecutive missing data and demonstrates better adaptability to datasets with consecutive missing values.

We conduct experiments on the Electricity dataset with varying lengths of consecutive missing data, and the results are shown in Figure 3a. As the length of consecutive missing data increases, the performance of all imputation methods declines to varying degrees. Models that perform reasonably well under low levels of consecutive missing data exhibit poor performance at higher levels. In contrast, our method is less impacted

Table 2: Imputation results on real datasets with different missing rates and a fixed 50 missing length. The label "_F" indicates imputation using forecasting methods.

		LSTI		Transformer		CSDI		SAITS		TimesNet		Imputeformer		TimesNet_F		iTransformer_F	
	R_m	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
Electricity	10%	**0.224**	**0.108**	0.775	0.896	0.653	0.765	0.769	0.907	0.607	0.687	0.752	0.860	0.290	0.181	0.286	0.192
	30%	**0.255**	**0.144**	0.806	0.961	0.658	0.766	0.809	0.982	0.686	0.844	0.793	0.920	0.375	0.296	0.395	0.339
	50%	**0.269**	**0.157**	0.802	0.971	0.770	0.980	0.809	0.987	0.698	0.876	0.795	0.932	0.478	0.431	0.524	0.504
	70%	**0.321**	**0.209**	0.863	1.121	0.766	0.979	0.821	1.019	0.733	0.945	0.803	0.940	0.578	0.572	0.637	0.668
Traffic	10%	**0.265**	**0.308**	0.781	1.246	0.601	0.896	0.784	1.238	0.707	1.078	0.755	1.160	0.323	0.392	0.288	0.343
	30%	**0.301**	**0.341**	0.762	1.164	0.555	0.871	0.769	1.166	0.697	1.056	0.756	1.170	0.389	0.478	0.373	0.454
	50%	**0.292**	**0.322**	0.761	1.163	0.610	0.913	0.773	1.181	0.782	1.255	0.753	1.139	0.514	0.689	0.541	0.745
	70%	**0.366**	**0.428**	0.766	1.166	0.619	0.961	0.767	1.156	0.826	1.397	0.773	1.244	0.604	0.833	0.629	0.877
METR-LA	10%	**0.478**	**0.710**	0.770	1.099	0.509	0.938	0.704	1.266	0.594	0.988	0.690	1.003	0.638	1.249	0.577	0.928
	30%	**0.508**	**0.747**	0.795	1.272	0.606	1.345	0.742	1.381	0.559	0.926	0.804	1.382	0.592	0.951	0.612	0.987
	50%	**0.568**	**0.867**	0.744	1.210	0.626	1.375	0.727	1.351	0.589	1.048	0.788	1.344	0.674	1.150	0.672	1.093
	70%	**0.621**	**1.025**	0.764	1.301	0.653	1.396	0.754	1.450	0.629	1.156	0.799	1.386	0.716	1.219	0.725	1.188
Guangzhou	10%	**0.410**	**0.355**	0.766	1.024	0.557	0.607	0.800	1.107	0.856	1.246	0.712	0.867	0.443	0.399	0.481	0.491
	30%	**0.395**	**0.321**	0.750	0.920	0.581	0.664	0.752	0.971	0.893	1.293	0.787	0.971	0.527	0.533	0.588	0.633
	50%	**0.428**	**0.368**	0.733	0.904	0.563	0.631	0.766	1.007	0.888	1.306	0.757	0.950	0.595	0.639	0.637	0.728
	70%	**0.491**	**0.457**	0.747	0.928	0.631	0.736	0.775	0.990	0.860	1.238	0.779	0.995	0.692	0.813	0.713	0.847
PEMS04	10%	**0.232**	**0.109**	0.819	0.915	0.447	0.319	0.780	0.834	0.728	0.861	0.854	0.965	0.281	0.160	0.366	0.279
	30%	**0.267**	**0.142**	0.827	0.897	0.553	0.505	0.762	0.813	0.711	0.865	0.845	0.969	0.531	0.611	0.531	0.532
	50%	**0.349**	**0.235**	0.812	0.901	0.622	0.628	0.799	0.895	0.719	0.921	0.863	0.994	0.743	1.040	0.737	0.940
	70%	**0.478**	**0.388**	0.821	0.926	0.639	0.630	0.870	1.045	0.776	0.998	0.956	1.133	0.809	1.056	0.893	1.264

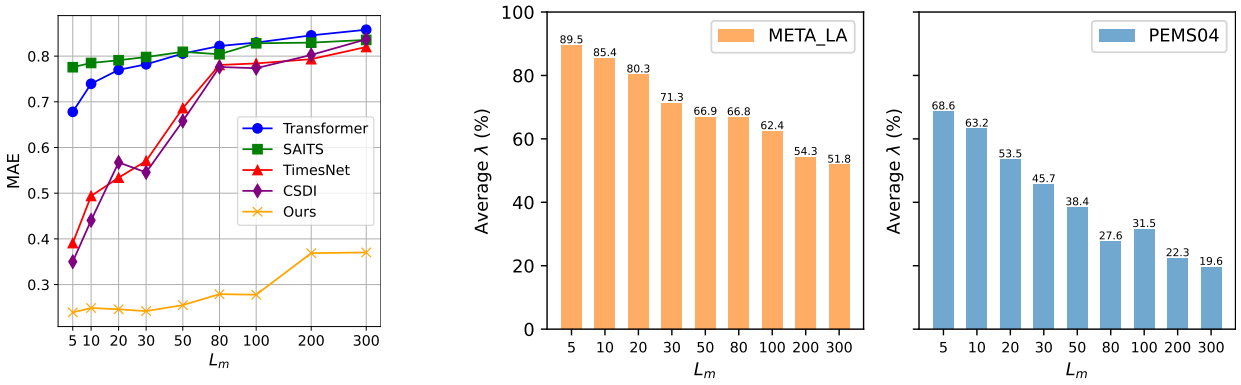
by the length of consecutive missing data, exhibiting more stable performance across different lengths. It performs well under both low and high levels of consecutive missing data.

Furthermore, we observe that some models reach their worst performance even with very short lengths of consecutive missing data, whereas they perform much better under MCAR. This indicates that different models have varying levels of robustness to consecutive missing data. Some models experience a significant performance drop with only a small amount of consecutive missing data, which poses a substantial risk in real-world scenarios. However, our method demonstrates excellent robustness to consecutive missing data lengths and can effectively handle large-scale missing data problems in real-world applications.

4.2.2 Varying missing rate

In addition to handling data with different lengths of consecutive missing values, the imputation model should also adapt to data with varying missing rates. Table 2 presents the experimental results under different missing rates with a fixed missing length of 50. As shown, our method outperforms all baselines in every case. For instance, regarding the MSE metric, our method leads to an average improvement of 74.62%, 57.81%, 28.19%, 52.38%, 70.15% compare to other methods on Electricity, Traffic, METR-LA, Guangzhou, and PEMS04 datasets, respectively. This demonstrates that our model has strong robustness and can adapt to various types of missing data.

Additionally, we observe an anomalous phenomenon in some models where increasing the missing rate on certain datasets actually improves the results. This occurs because, at lower missing rates, the randomly consecutive missing values may be unevenly distributed, causing instability in the imputation models. Different random missing data can produce significantly varying results. If the missing intervals happen to concentrate within the model's imputation window, the model has to impute at a higher missing rate within that interval, leading to unexpectedly poor results.



(a) MAE on Electricity.

(b) λ on META_LA and PEMS04.

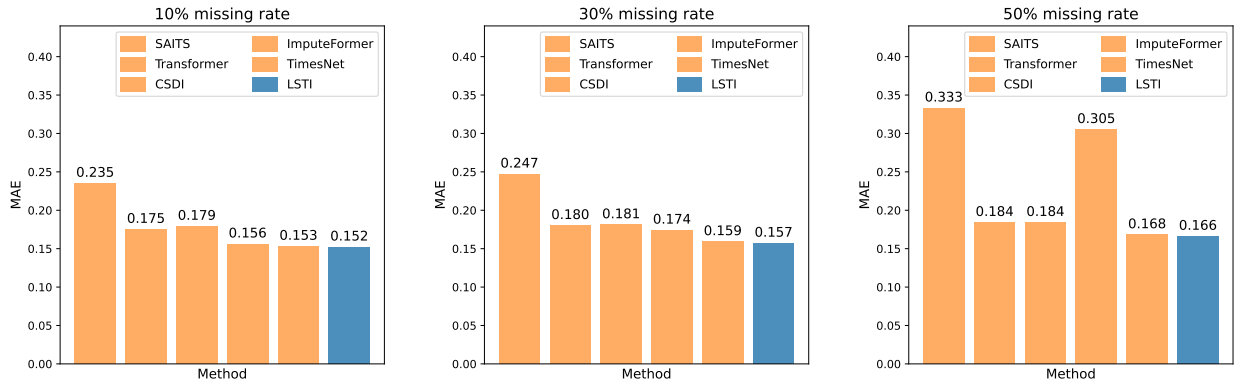
Figure 3: Results of MAE and average λ with varying lengths of consecutive missing data.

4.3 Analysis λ

We test the output λ of the Meta-weighting module on two datasets and record the average values of all λ during testing. The relevant results are shown in Figure 3b. As can be seen, λ generally decreases as L_m increases, indicating that the Short-Term Imputer is more dominant in datasets with shorter L_m, whereas the Long-Term Imputer is more dominant in datasets with longer L_m. This is because shorter consecutive missing data can more easily capture information near the missing values, making short-term dependencies more effective for imputation. Conversely, longer consecutive missing data can only obtain useful information from distant points, necessitating the use of long-term dependencies for better imputation. This demonstrates that our Meta-weighting module effectively learns the importance of long-term and short-term dependencies in different missing datasets and can adaptively adjust the weights based on the characteristics of the data.

4.4 Analysis data missing type

In real-world scenarios, various missing data patterns can occur. The straightforward Blackout missing pattern fails to fully capture real-world scenarios. In this section, we explore and conduct experiments under various missing patterns to assess the adaptability and effectiveness of LSTI across different settings.

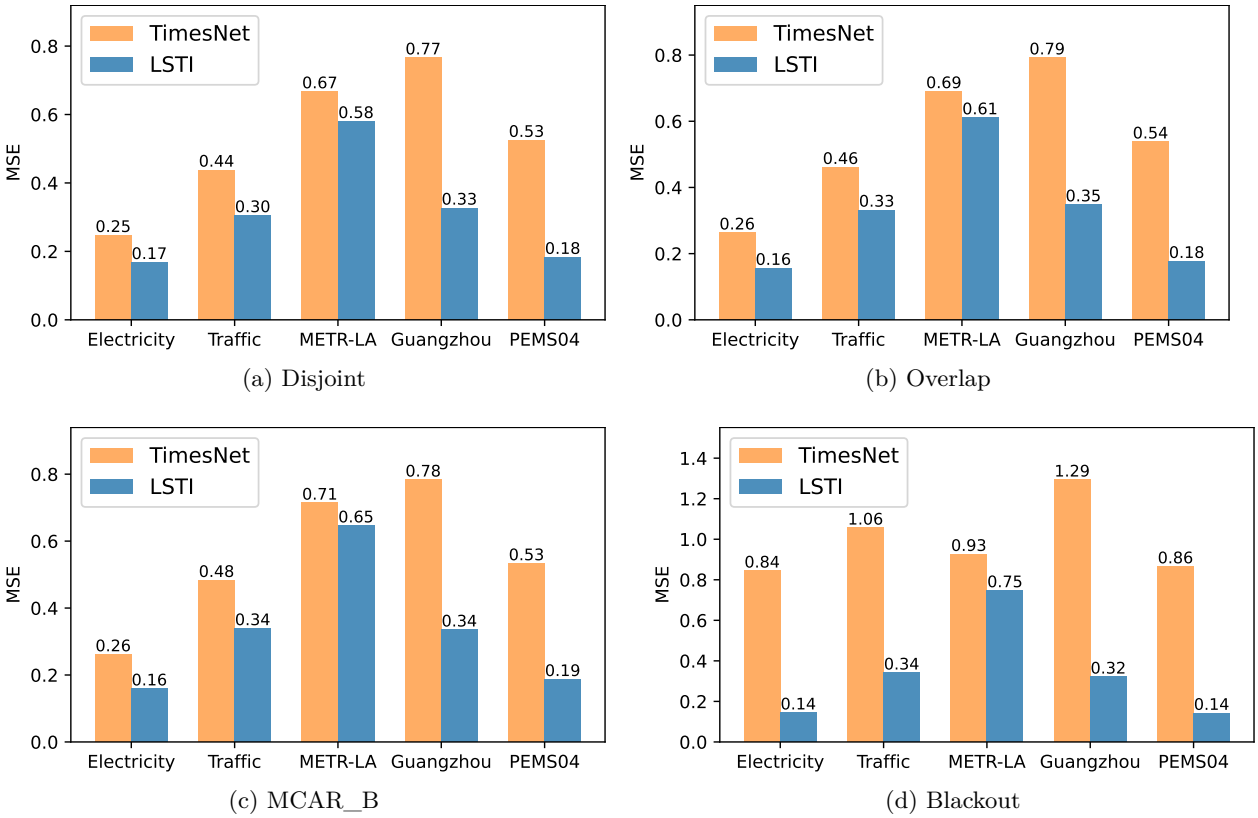


(a) 10% missing rate

(b) 30% missing rate

(c) 50% missing rate

Figure 4: Experimental results of various methods on the PEMS04 dataset under the MCAR missing type.



(a) Disjoint (b) Overlap

(c) MCAR_B (d) Blackout

Figure 5: Comparison of MSE between TimesNet and LSTI across different datasets and missing patterns. All experiments were conducted with L_m=50 and R_m=30% settings.

4.4.1 Analysis MCAR

Missing Completely At Random (MCAR) is a widely adopted missing data pattern in time-series imputation. It describes a scenario where the occurrence of missing data is independent of the data itself or any other variables. Under the MCAR pattern, missing data points are typically distributed uniformly across the dataset. Figure 4 illustrates the experimental results of various methods on the PEMS04 dataset under the MCAR missing pattern, with all models trained using the same MCAR configuration. As shown, despite not being specifically designed for MCAR, LSTI consistently achieves superior performance across different missing rates. This highlights the robustness and versatility of LSTI.

4.4.2 Analysis other consecutive missing types

To evaluate the performance of LSTI under different consecutive missing patterns, we follow the scheme in Khayati et al. (2020), which categorizes these patterns into four types: 'Disjoint,' 'Overlap,' 'MCAR_B,' and 'Blackout.' Specifically, the 'Disjoint' pattern requires the missing blocks to have as little overlap as possible, while the 'Overlap' pattern demands the missing blocks to have as much overlap as possible. The 'MCAR_B' pattern involves randomly selecting missing blocks for each channel independently, whereas the 'Blackout' pattern randomly selects identical missing blocks across all channels. All experiments were conducted with L_m=50 and R_m=30% settings.

The experimental results are shown in Figure 5. It can be seen that our LSTI outperforms TimesNet across all datasets and missing patterns, demonstrating that our method has exceptional adaptability to various continuous missing patterns and exhibits high robustness, showing minimal sensitivity to the specific missing pattern. The figure also reveals that while TimesNet's performance remains consistent across the Disjoint, Overlap, and MCAR_B missing patterns, it significantly deteriorates under the Blackout missing pattern.

Table 3: Ablation study results of different LSTI modules, with L_m values of [10, 30, 50, 100, 300], and all results averaged over these five different L_m values.

					META-LA		PEMS04	
fL	bL	L	S	LS	MAE	MSE	MAE	MSE
✓	-	-	-	-	0.794	1.423	0.437	0.376
-	✓	-	-	-	0.692	1.362	0.413	0.367
✓	✓	✓	-	-	0.611	0.973	0.380	0.324
-	-	-	✓	-	0.558	0.963	0.476	0.435
✓	✓	✓	✓	✓	**0.495**	**0.721**	**0.347**	**0.243**

This indicates that traditional imputation methods are less adaptable to the Blackout missing pattern, where missing values span all channels, making inference from other channels infeasible. In contrast, our LSTI model effectively imputes data even under the challenging Blackout pattern, addressing a key limitation of traditional imputation approaches.

4.5 Ablation study

To investigate the impact of different LSTI modules on imputation performance, we conduct an ablation study on LSTI with the experimental results presented in Table 3. The abbreviations fL, bL, L, S, and LS represent the following configurations: only the forward prediction network, only the backward prediction network, only the Long-Term Imputer, only the Short-Term Imputer, and the complete LSTI model, respectively. We find that in all cases, the Long-Term Imputer outperformed both the forward and backward prediction networks when used individually. This suggests that the strategy of weighted fusion of predictions made before and after the missing interval is both practical and highly effective.

Furthermore, while the Short-Term Imputer generally performed worse than the Long-Term Imputer on most datasets, their combined results achieved optimal performance. This is attributed to the fact that the Short-Term Imputer performs better when L_m is small, but its performance significantly declines as L_m increases. However, due to the presence of the Meta-weighting module, the negative impact of the Short-Term Imputer is minimized, resulting in the final averaged outcome being optimal. This demonstrates the importance of integrating both short-term and long-term components in the LSTI model for achieving robust imputation outcomes.

5 Conclusion

Long-interval consecutive missing data is a common issue in real-world time series datasets, which traditional deep learning imputation methods struggle to address effectively. To tackle this, we propose the Long Short-Term Imputer (LSTI), consisting of a Long-Term Imputer, a Short-Term Imputer, and a Meta-weighting module. This model can capture both long-term and short-term dependencies in time series and performs weighted merging based on data characteristics. Experiments on five real-world datasets demonstrate that our method can effectively impute long-interval consecutive missing data.

Broader Impact Statement

Our work proposes a method for imputing time series data with long-interval consecutive missing values, which can indirectly improve the quality of downstream tasks but is not directly applicable to real-world tasks. Furthermore, by combining forecasting and imputation models, this research can guide future studies in machine learning and data science, promoting the development of more sophisticated and adaptive imputation methods. Therefore, our paper is primarily focused on scientific research and has no obvious negative social impact.

References

Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. Brits: Bidirectional recurrent imputation for time series. *Advances in neural information processing systems*, 31, 2018.

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.

Chao Chen, Karl Petty, Alexander Skabardonis, Pravin Varaiya, and Zhanfeng Jia. Freeway performance measurement system: mining loop detector data. *Transportation research record*, 1748(1):96–102, 2001.

Xinyu Chen, Yixian Chen, and Zhaocheng He. Urban traffic speed dataset of guangzhou, china. *March 22, 2018. Distributed by Zenodo.*, 2018.

Zeng Chen, Huan Xu, Peng Jiang, Shanen Yu, Guang Lin, Igor Bychkov, Alexey Hmelnov, Gennady Ruzhnikov, Ning Zhu, and Zhen Liu. A transfer learning-based lstm strategy for imputing large-scale consecutive missing data and its application in a water quality prediction system. *Journal of Hydrology*, 602:126573, 2021.

Wenjie Du, David Côté, and Yan Liu. Saits: Self-attention-based imputation for time series. *Expert Systems with Applications*, 219:119619, 2023.

Xiaolong Guo, Shijia Zhu, Zhiwei Yang, Hao Liu, and Tianshu Bi. Consecutive missing data recovery method based on long-short term memory network. In *2021 3rd Asia Energy and Electrical Engineering Symposium (AEEES)*, pp. 988–992. IEEE, 2021.

Ashish Gupta, Hari Prabhat Gupta, Bhaskar Biswas, and Tanima Dutta. An early classification approach for multivariate time series of on-vehicle sensors in transportation. *IEEE Transactions on Intelligent Transportation Systems*, 21(12):5316–5327, 2020.

Zahra Hajihashemi and Mihail Popescu. A multidimensional time-series similarity measure with applications to eldercare monitoring. *IEEE Journal of biomedical and health informatics*, 20(3):953–962, 2015.

Md Kamrul Hasan, Md Ashraful Alam, Shidhartho Roy, Aishwariya Dutta, Md Tasnim Jawad, and Sunanda Das. Missing value imputation affects the performance of machine learning: A review and analysis of the literature (2010–2021). *Informatics in Medicine Unlocked*, 27:100799, 2021.

Mourad Khayati, Alberto Lerner, Zakhar Tymchenko, and Philippe Cudré-Mauroux. Mind the gap. *Proceedings of the VLDB Endowment*, 13:768–782, 2020.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 95–104, 2018.

Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in Neural Information Processing Systems*, 35:9881–9893, 2022.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.

Yukai Liu, Rose Yu, Stephan Zheng, Eric Zhan, and Yisong Yue. Naomi: Non-autoregressive multiresolution sequence imputation. *Advances in neural information processing systems*, 32, 2019.

Yonghong Luo, Ying Zhang, Xiangrui Cai, and Xiaojie Yuan. E2gan: End-to-end generative adversarial network for multivariate time series imputation. In *Proceedings of the 28th international joint conference on artificial intelligence*, pp. 3094–3100. AAAI Press Palo Alto, CA, USA, 2019.

Jun Ma, Jack CP Cheng, Yuexiong Ding, Changqing Lin, Feifeng Jiang, Mingzhu Wang, and Chong Zhai. Transfer learning for long-interval consecutive missing values imputation without external features in air pollution time series. *Advanced Engineering Informatics*, 44:101092, 2020.

Metro. Metr-la dataset. https://drive.google.com/drive/folders/10FOTa6HXPqX8Pf5WRoRwcFnW9BrNZEIX.

Tong Nie, Guoyang Qin, Wei Ma, Yuewen Mei, and Jian Sun. Imputeformer: Low rankness-induced transformers for generalizable spatiotemporal imputation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2260–2271, 2024.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2022.

PeMS. Traffic dataset. http://pems.dot.ca.gov/.

Adityo Prakosa, Maxime Sermesant, Hervé Delingette, Stéphanie Marchesseau, Eric Saloux, Pascal Allain, Nicolas Villain, and Nicholas Ayache. Generation of synthetic but visually realistic time series of cardiac images combining a biophysical model and clinical images. *IEEE transactions on medical imaging*, 32(1): 99–109, 2012.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3):1181–1191, 2020.

Siyuan Shan, Yang Li, and Junier B Oliva. Nrtsi: Non-recurrent time series imputation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023.

Ikaro Silva, George Moody, Daniel J Scott, Leo A Celi, and Roger G Mark. Predicting in-hospital mortality of icu patients: The physionet/computing in cardiology challenge 2012. In *2012 Computing in Cardiology*, pp. 245–248. IEEE, 2012.

Akin Tascikaraoglu, Borhan M Sanandaji, Gianfranco Chicco, Valeria Cocina, Filippo Spertino, Ozan Erdinc, Nikolaos G Paterakis, and João PS Catalão. Compressive spatio-temporal forecasting of meteorological quantities and photovoltaic power. *IEEE Transactions on Sustainable Energy*, 7(3):1295–1305, 2016.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34: 24804–24816, 2021.

UCI. Electricity dataset. https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Chantha Wongoutong. Imputation for consecutive missing values in non-stationary time series data. *Advances and Applications in Statistics*, 64(1):87–102, 2020.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34: 22419–22430, 2021.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

Xiuwen Yi, Yu Zheng, Junbo Zhang, and Tianrui Li. St-mvl: filling missing values in geo-sensory time series data. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, 2016.

Jinsung Yoon, William R Zame, and Mihaela van der Schaar. Estimating missing data in temporal data streams using multi-directional recurrent neural networks. *IEEE Transactions on Biomedical Engineering*, 66(5):1477–1490, 2018.

Yu Zheng, Xiuwen Yi, Ming Li, Ruiyuan Li, Zhangqing Shan, Eric Chang, and Tianrui Li. Forecasting fine-grained air quality based on big data. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 2267–2276, 2015.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11106–11115, 2021.

A Standard deviation

Table 4: Standard deviation on real datasets with different missing lengths and a fixed 30% missing rate. The label "_F" indicates models using forecasting methods.

		LSTI		CSDI		TimesNet		SAITS		Transformer		TimesNet_F		iTransformer_F	
	L_m	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
Electricity	10	0.008	0.011	0.094	0.145	0.031	0.059	0.005	0.007	0.018	0.038	0.017	0.032	0.007	0.018
	30	0.006	0.010	0.021	0.028	0.038	0.072	0.009	0.025	0.023	0.057	0.001	0.002	0.011	0.009
	50	0.003	0.007	0.023	0.033	0.032	0.082	0.011	0.025	0.014	0.035	0.002	0.010	0.023	0.026
	100	0.022	0.028	0.027	0.047	0.025	0.037	0.014	0.016	0.013	0.036	0.026	0.029	0.029	0.021
	300	0.025	0.025	0.014	0.045	0.027	0.137	0.002	0.019	0.012	0.032	0.033	0.056	0.033	0.056
Traffic	10	0.005	0.016	0.003	0.011	0.015	0.018	0.012	0.040	0.014	0.041	0.021	0.060	0.017	0.044
	30	0.008	0.019	0.025	0.052	0.030	0.059	0.014	0.036	0.013	0.043	0.020	0.071	0.018	0.046
	50	0.028	0.038	0.025	0.050	0.037	0.099	0.015	0.064	0.014	0.067	0.042	0.068	0.018	0.026
	100	0.023	0.031	0.008	0.046	0.015	0.073	0.011	0.051	0.012	0.047	0.048	0.081	0.049	0.086
	300	0.031	0.073	0.042	0.108	0.013	0.054	0.010	0.042	0.007	0.037	0.039	0.075	0.041	0.083
METR-LA	10	0.024	0.058	0.070	0.288	0.034	0.078	0.116	0.397	0.100	0.254	0.042	0.155	0.051	0.093
	30	0.018	0.061	0.076	0.274	0.026	0.100	0.046	0.179	0.044	0.158	0.057	0.231	0.032	0.043
	50	0.004	0.027	0.053	0.215	0.055	0.207	0.037	0.115	0.013	0.075	0.023	0.115	0.044	0.181
	100	0.115	0.352	0.122	0.450	0.050	0.222	0.092	0.261	0.087	0.297	0.027	0.100	0.046	0.217
	300	0.131	0.280	0.188	0.790	0.102	0.370	0.193	0.657	0.179	0.642	0.182	0.602	0.171	0.591
Guangzhou	10	0.006	0.011	0.065	0.078	0.030	0.062	0.018	0.052	0.034	0.061	0.023	0.032	0.014	0.017
	30	0.050	0.082	0.086	0.121	0.019	0.075	0.021	0.092	0.021	0.064	0.026	0.049	0.007	0.026
	50	0.007	0.028	0.031	0.074	0.082	0.168	0.025	0.073	0.021	0.041	0.023	0.038	0.037	0.029
	100	0.053	0.055	0.032	0.063	0.008	0.014	0.013	0.032	0.010	0.040	0.077	0.121	0.069	0.110
	300	0.009	0.023	0.036	0.026	0.106	0.294	0.033	0.084	0.019	0.047	0.014	0.012	0.016	0.009
PEMS04	10	0.006	0.011	0.024	0.025	0.005	0.006	0.016	0.027	0.008	0.012	0.118	0.238	0.028	0.108
	30	0.023	0.015	0.052	0.082	0.029	0.053	0.051	0.085	0.073	0.117	0.056	0.223	0.032	0.205
	50	0.027	0.037	0.037	0.076	0.024	0.072	0.027	0.050	0.026	0.044	0.029	0.094	0.028	0.035
	100	0.061	0.098	0.049	0.108	0.035	0.054	0.047	0.084	0.040	0.070	0.124	0.236	0.118	0.181
	300	0.080	0.138	0.042	0.100	0.027	0.103	0.018	0.065	0.011	0.018	0.039	0.059	0.026	0.037

Table 5: Standard deviation on real datasets with different missing rates and a fixed 50 missing length. The label "_F" indicates imputation using forecasting methods.

		LSTI		CSDI		TimesNet		SAITS		Transformer		TimesNet_F		iTransformer_F	
	R_m	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
Electricity	10%	0.003	0.003	0.092	0.158	0.030	0.055	0.023	0.052	0.034	0.079	0.014	0.017	0.012	0.015
	30%	0.003	0.007	0.038	0.095	0.032	0.082	0.011	0.025	0.014	0.035	0.002	0.010	0.023	0.026
	50%	0.007	0.011	0.044	0.101	0.016	0.049	0.008	0.012	0.031	0.053	0.021	0.036	0.006	0.015
	70%	0.018	0.023	0.035	0.070	0.018	0.033	0.007	0.016	0.023	0.046	0.020	0.038	0.015	0.027
Traffic	10%	0.014	0.009	0.084	0.179	0.049	0.115	0.021	0.049	0.020	0.061	0.033	0.051	0.052	0.074
	30%	0.028	0.038	0.025	0.050	0.037	0.099	0.015	0.064	0.014	0.067	0.042	0.068	0.018	0.026
	50%	0.012	0.018	0.075	0.105	0.026	0.088	0.011	0.051	0.016	0.055	0.070	0.162	0.025	0.060
	70%	0.029	0.046	0.065	0.109	0.018	0.029	0.007	0.014	0.014	0.021	0.023	0.046	0.027	0.050
METR-LA	10%	0.164	0.393	0.071	0.246	0.171	0.534	0.191	0.653	0.171	0.525	0.238	0.767	0.144	0.469
	30%	0.004	0.027	0.053	0.215	0.055	0.207	0.037	0.115	0.013	0.075	0.023	0.115	0.044	0.181
	50%	0.050	0.164	0.046	0.181	0.034	0.105	0.037	0.174	0.014	0.101	0.029	0.070	0.048	0.168
	70%	0.034	0.134	0.040	0.178	0.041	0.096	0.003	0.020	0.016	0.024	0.046	0.145	0.027	0.039
Guangzhou	10%	0.020	0.048	0.090	0.148	0.039	0.089	0.048	0.133	0.048	0.110	0.013	0.041	0.054	0.134
	30%	0.007	0.028	0.031	0.074	0.082	0.168	0.025	0.073	0.021	0.041	0.023	0.038	0.037	0.029
	50%	0.019	0.026	0.040	0.075	0.024	0.043	0.022	0.053	0.007	0.024	0.029	0.048	0.015	0.041
	70%	0.008	0.002	0.016	0.053	0.036	0.114	0.006	0.005	0.011	0.017	0.039	0.091	0.022	0.038
PEMS04	10%	0.056	0.042	0.046	0.053	0.043	0.094	0.032	0.049	0.046	0.093	0.037	0.024	0.027	0.040
	30%	0.027	0.037	0.037	0.076	0.024	0.072	0.027	0.050	0.026	0.044	0.029	0.094	0.028	0.035
	50%	0.046	0.072	0.065	0.103	0.047	0.136	0.032	0.056	0.033	0.049	0.055	0.280	0.036	0.121
	70%	0.008	0.013	0.067	0.102	0.035	0.055	0.049	0.135	0.007	0.008	0.072	0.227	0.039	0.138