ImputeINR: Time Series Imputation via Implicit Neural Representations for Disease Diagnosis with Missing Data

Mengxuan Li[1,2], **Ke Liu**[1], **Jialong Guo**[1],
Jiajun Bu[1], **Hongwei Wang**[2,*], **Haishuai Wang**[1,*]
[1]Zhejiang Key Laboratory of Accessible Perception and Intelligent Systems,
College of Computer Science, Zhejiang University, China
[2] The Zhejiang University-University of Illinois Urbana-Champaign Institute, Zhejiang University
{mengxuanli, hongweiwang}@intl.zju.edu.cn, {keliu99, jialongguo, bjj, haishuai.wang}@zju.edu.cn

Abstract

Healthcare data frequently contain a substantial proportion of missing values, necessitating effective time series imputation to support downstream disease diagnosis tasks. However, existing imputation methods focus on discrete data points and are unable to effectively model sparse data, resulting in particularly poor performance for imputing substantial missing values. In this paper, we propose a novel approach, ImputeINR, for time series imputation by employing implicit neural representations (INR) to learn continuous functions for time series. ImputeINR leverages the merits of INR in that the continuous functions are not coupled to sampling frequency and have infinite sampling frequency, allowing ImputeINR to generate fine-grained imputations even on extremely sparse observed values. Extensive experiments conducted on eight datasets with five ratios of masked values show the superior imputation performance of ImputeINR, especially for high missing ratios in time series data. We also validate that applying ImputeINR to impute missing values in healthcare data enhances the performance of downstream disease diagnosis tasks.

1 Introduction

Healthcare data inherently exhibit a temporal structure, as they are often collected sequentially over time in the form of physiological signals, electronic health records, and clinical monitoring data, making time series analysis a fundamental approach for understanding and predicting health-related outcomes [Schaffer *et al.*, 2021; Wang *et al.*, 2018; Li *et al.*, 2023a; Li *et al.*, 2023b]. However, real-world healthcare data are frequently compromised by a substantial amount of missing values, which arise from various sources, such as sensor malfunctions, transmission errors, or irregular reporting intervals. The presence of missing values poses a significant challenge to disease diagnosis, as incomplete or corrupted data can distort model training and lead to unreliable predictions. Consequently, the imputation of missing data becomes a critical step in disease diagnosis. By filling in the gaps, imputation enables the restoration of a complete dataset, allowing diagnostic models to perform without the biases or inaccuracies that would otherwise result from missing information.

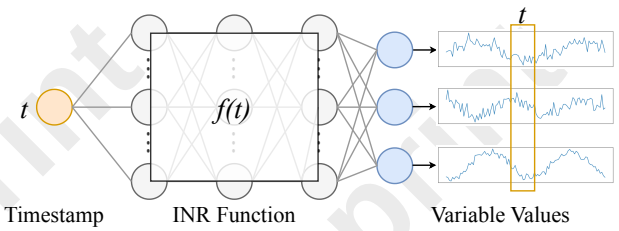

Figure 1: The illustration of INR applied to time series data.

However, most existing imputation methods do not highlight the cases of extremely sparse observed values. These works assume that the proportion of masked values requiring imputation does not exceed 50%, which means that these methods still require a certain amount of known information. But in real-world healthcare data, the proportion of missing values is likely to be even higher. For example, the MIMIC-III dataset [Johnson *et al.*, 2018], a public health-related database, contains 63.15% missing values. The PhysioNet 2012 and 2019 challenges [Silva *et al.*, 2012; Reyna *et al.*, 2019], which involved intensive care unit (ICU) patient data, reported a missing data rate of 79.67%, highlighting the prevalence of high missing rates in real-world healthcare datasets. Although some recent studies try to address imputation tasks in sparse scenarios [Alcaraz and Strodthoff, 2022; Tashiro *et al.*, 2021], most work relies on diffusion-based methods, which involve the simulation of high-dimensional stochastic processes and multi-step iterative procedures. Consequently, these approaches are typically time-consuming and require a large number of training iterations, and their limited effectiveness in achieving accurate imputation can significantly impair the performance of downstream disease diagnosis tasks. How to perform imputation in extremely sparse scenarios remains challenging.

Recently, implicit neural representation (INR) has emerged as an effective method for continuously encoding diverse signals [Liu *et al.*, 2023b; Molaei *et al.*, 2023]. As shown in Fig-

*Corresponding authors.

ure 1, it learns continuous functions from discrete data points, mapping coordinates to signal values. By representing complex structures in a compact form, INR is not coupled to sampling frequency anymore, which allows for multi-sampling frequency inputs enabling effective feature extraction even with absent observed samples. Additionally, as a continuous function, INR has infinite sampling frequency, which means it can be queried at any coordinate. This capacity for infinite sampling frequency interpolation sets it apart from other imputation methods, making it a promising approach for fine-grained imputation.

In this paper, we propose a novel time series imputation approach, named ImputeINR, which achieves effective imputation even in extremely sparse scenarios and the imputed data can further enhance the performance of downstream disease diagnosis tasks. Generally, we learn an INR continuous function for the target time series data and enable fine-grained interpolation with infinite sampling frequency. The parameters of the INR function are predicted by a transformer-based feed-forward network conditioned with sparsely observed values. More specifically, an adaptive group-based form of the INR function is proposed for capturing complex temporal patterns and cross-channel correlation features. It is a multilayer perceptron (MLP) composed of global layers and group layers. The former focuses on correlation information across all channels, while the latter emphasizes correlation information among variables within a single group. We observe that INR exhibits the strongest representational capacity when partitioning variables with similar distributions into the same group. Therefore, variable clustering is applied to determine the variable partition. Additionally, a multi-scale feature extraction module is incorporated to capture patterns at various temporal scales, achieving better fine-grained imputation. Experimentally, ImputeINR achieves state-of-the-art performance on eight imputation benchmarks with various ratios of masked values. Moreover, we demonstrate that the imputed data with ImputeINR, compared to other methods, significantly improves the performance of downstream disease diagnosis tasks. The major contributions of this paper are summarized as follows:

- We propose ImputeINR, which learns INR continuous function to represent the continuous time series data. It achieves effective imputation in extremely sparse scenarios and the imputed data can improve the performance of downstream disease diagnosis tasks.
- We design an adaptive group-based form of the INR continuous function to effectively capture intricate temporal patterns and cross-channel correlation features.
- We apply variable clustering to determine the variable partition, allowing our group-based architecture to learn correlation information across all variables and among variables with similar distributions.
- Extensive experiments show that ImputeINR outperforms other baselines on eight datasets under five ratios of masked values. We also demonstrate that using ImputeINR for healthcare data imputation significantly enhances disease diagnosis performance.

2 Related Work

2.1 Time Series Imputation

The earliest time series imputation methods are based on the statistical properties of the data, using mean/median values or statistical models to fill in missing values, such as SimpleMean/SimpleMedian [Fung, 2006] and ARIMA [Afrifa-Yamoah *et al.*, 2020]. Then, machine learning methods learn data patterns, showing greater adaptability and accuracy. Prominent works of these approaches include KNNI [Altman, 1992] and MICE [Van Buuren and Groothuis-Oudshoorn, 2011]. Although these methods are easy to interpret, their limitations lie in capturing the complex temporal and variable information inherent in time series data. Recently, there has been widespread interest in using deep models to capture complex temporal patterns for imputation of missing values, due to their powerful representation capabilities. Common architectures include RNN-based methods (e.g., M-RNN [Yoon *et al.*, 2018], NRTSI [Shan *et al.*, 2023], and BRITS [Cao *et al.*, 2018]) , CNN-based methods (e.g., TimesNet [Wu *et al.*, 2023]), MLP-based methods (e.g., DLinear [Zeng *et al.*, 2023], TimeMixer [Wang *et al.*, 2024]), transformer-based methods (e.g., SAITS [Du *et al.*, 2023], FPT [Zhou *et al.*, 2023], iTransformer [Liu *et al.*, 2024], ImputeFormer [Nie *et al.*, 2024]), diffusion-based methods (e.g., CSDI [Tashiro *et al.*, 2021], SSSD [Alcaraz and Strodthoff, 2022]).

2.2 Implicit Neural Representations

INR uses neural networks to model signals as continuous functions rather than explicitly representing them as discrete points. It captures complex high-dimensional patterns in data by learning a continuous mapping from input coordinates to output values. Various scenarios have seen successful applications, such as 2D image generation [Saragadam *et al.*, 2022; Liu *et al.*, 2023a; Zhang *et al.*, 2024], 3D scene reconstruction [Yin *et al.*, 2022; Liu *et al.*, 2023b; Yang *et al.*, 2024], and video representations [Mai and Liu, 2022; Guo *et al.*, 2025]. Since INR learns a continuous function, it is not coupled to the resolution, which implies that the memory needed to parameterize the signal does not depend on spatial resolution but rather increases with the complexity of the underlying signal. Also, INR has infinite resolution, which means it can be sampled at an arbitrary sampling frequency. Therefore, we leverage this characteristic of INR to perform time series imputation tasks. Sampling from the continuous function of INR enables fine-grained imputation even with extremely sparse observed data. To learn the INR for target signal, there are mainly two typical strategies: gradient-based meta-learning methods [Lee *et al.*, 2021; Liu *et al.*, 2023a] and feed-forward hyper-network prediction methods [Chen and Wang, 2022; Zhang *et al.*, 2024]. In this work, we use a transformer-based feed-forward method to predict the INR for time series data since it can be easily adopted to an end-to-end imputation framework.

2.3 Implicit Neural Representations on Time Series Data

Several works have explored the use of INR to model time series data. INRAD [Jeong and Shin, 2022] leverages INR

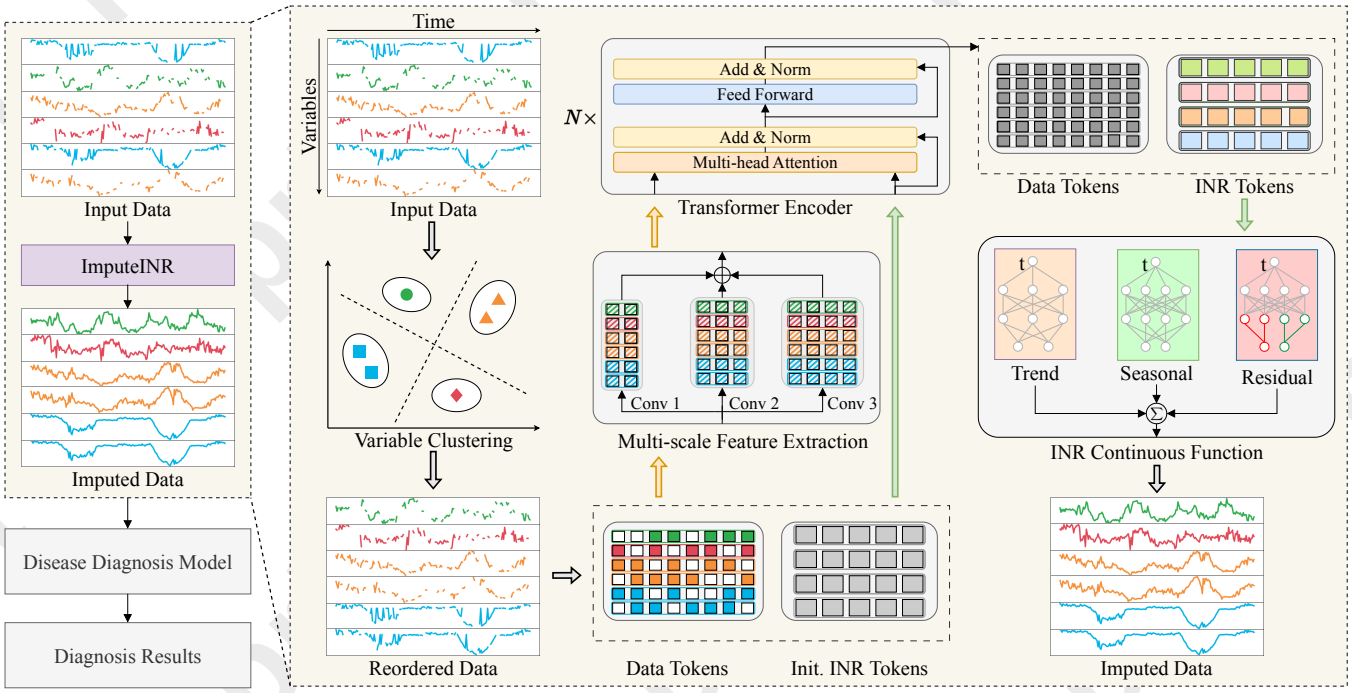

Figure 2: The overall workflow of the proposed method. The input data with missing values is imputed using the ImputeINR model, and the imputed data is then fed into the disease diagnosis model to obtain the diagnostic results. In ImputeINR, the INR tokens are predicted using a transformer encoder. These tokens serve as the parameters for the INR continuous function, which takes the timestamp t as input.

to reconstruct time series data for anomaly detection. TSINR [Li *et al.*, 2024] takes advantage of the spectral bias property of INR, prioritizing low-frequency signals and exhibiting reduced performance on high-frequency anomalous data to identify anomalies. However, these methods are specifically designed for anomaly detection tasks and are not suitable for imputation tasks. Currently, only a few works have attempted to leverage INR for time series imputation tasks, but each of these approaches has its own limitations. For example, HyperTime [Fons *et al.*, 2022] utilizes INR to learn a compressed latent representation to capture the underlying patterns in time series for imputation and generation. However, it uses a permutation-invariant set encoder to extract features, leading to an insufficient representation of the underlying patterns in the time series. In addition, TimeFlow [Naour *et al.*, 2023] utilizes continuous-time-dependent modeling and INR enhanced by a meta-learning-driven modulation mechanism for imputation and forecasting. However, treats each variable as an individual sample, thereby ignoring crucial inter-variable correlations. Moreover, it fits the entire time series at once becoming computationally inefficient, particularly when dealing with long time series.

3 Methodology

3.1 Problem Formulation

Imputation with Missing Data

Denote time series data with N variables and T timestamps as $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \in \mathbb{R}^{N \times T}$. The time series data $\mathbf{X}$ is incomplete and the mask rate is $r \in [0, 1]$. The corresponding binary mask matrix can be defined as $\mathbf{M} = \{m_{n,t}\} \in \{0, 1\}^{N \times T}$, where $m_{n,t} = 1$ if $x_{n,t}$ is observed, and $m_{n,t} = 0$ if $x_{n,t}$ is missing. The imputation task is to predict the missing values $\mathbf{X}_{\text{miss}}$ such that the predicted values $\hat{\mathbf{X}}$ satisfy $\hat{\mathbf{X}} = F_\theta(\mathbf{X}, \mathbf{M})$, where F_θ mentions the model with parameters θ. The goal is to minimize the reconstruction error between the masked data and the imputed data:

$$\mathcal{L}(\hat{\mathbf{X}}, \mathbf{X}_{\text{gt}}) = \frac{1}{|\mathbf{M}_{\text{miss}}|} \sum_{n=1}^{N} \sum_{t=1}^{T} (1 - m_{n,t}) \cdot (\hat{x}_{n,t} - x_{n,t})^2 , \tag{1}$$

where $|\mathbf{M}_{\text{miss}}|$ is the total number of missing values in $\mathbf{X}$ and $\mathbf{X}_{\text{gt}}$ is the ground truth.

Disease Diagnosis

Let $\mathbf{X}_{\text{imputed}}$ represent the imputed version of $\mathbf{X}$, where the missing values in $\mathbf{X}$ are imputed and the non-missing values in $\mathbf{X}$ are retained. The task is to utilize $\mathbf{X}_{\text{imputed}}$ for disease diagnosis, where a model $G(\cdot)$ is trained to predict disease labels $\hat{y} \in \{0, 1\}$ based on $\mathbf{X}_{\text{imputed}}$:

$$\hat{y} = G\left(\mathbf{X}_{\text{imputed}}\right), \tag{2}$$

where $\hat{y} = 0$ denotes normal, and $\hat{y} = 1$ indicates the presence of disease.

3.2 Method Overview

The core idea of ImputeINR is to leverage the inherent capability of INR to learn continuous functions, allowing for fine-grained imputation of time series data by querying at arbitrary timestamps. By leveraging this flexibility,

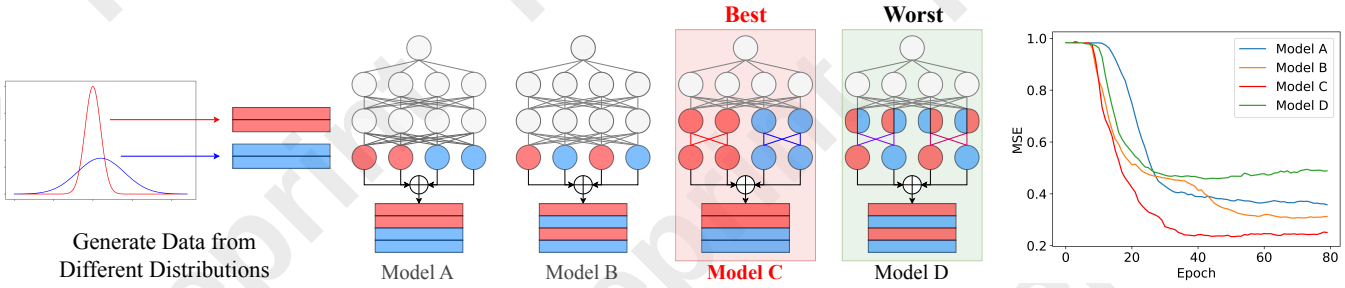

Figure 3: The four architectures we test to evaluate the representation capability of the INR continuous function for the synthetic time series dataset. The synthetic dataset consists of four variables, generated from two distinct distributions. The results prove that the representation capability of INR is strongest when variables from the same distribution are fitted by the same group.

ImputeINR can generate smooth and accurate estimates for missing data points, even with extremely sparse observed data. However, since time series data has inherently intricate temporal patterns and multi-variable properties, using a simple MLP as the INR continuous function to fit it is challenging. To address these issues, we design a novel form of INR continuous function specifically for time series data. Following the previous work [Fons *et al.*, 2022; Li *et al.*, 2024], this form includes three components to capture the trend, seasonal, and residual information to deal with the unique temporal patterns. Furthermore, to enhance the ability of ImputeINR to model multi-variable data, we propose an adaptive group-based architecture to learn complicated residual information. Each group focuses on variables with similar distributions. And we use a clustering algorithm to determine the variable partition. To further enhance the imputation capability of ImputeINR, we incorporate a multi-scale module to capture information at different scales, improving fine-grained imputation performance.

Figure 2 demonstrates the overall workflow of the proposed method. The masked data is first reordered based on the variable clustering results so that variables with similar distributions are placed adjacent to each other. This is to enable the subsequent representation of variables within the same cluster using the same group-based MLP in the INR continuous function. Then the reordered masked data is standardized and segmented into patches to prepare the data tokens. Simultaneously, we initialize the INR tokens, which are learnable vector parameters. The processed data tokens are input into convolutional layers of different scales to extract multi-scale features. Subsequently, these extracted features and the initialized INR tokens are fed together into the transformer encoder to predict the INR tokens. These INR tokens are essentially the parameters of the INR continuous function. Based on these parameters, the INR continuous function takes the timestamp t as input and predicts the masked values. Finally, the imputed data is then fed into the disease diagnosis model to obtain the diagnostic results.

3.3 Variable Clustering

We adopt a clustering algorithm $\mathcal{C}$ to cluster the variables of the time series data $\mathbf{X} \in \mathbb{R}^{N \times T}$ based on the similarity matrix $S \in \mathbb{R}^{N \times N}$, which partitions the variables into K clusters:

$$\mathcal{C} : \mathbb{R}^{N \times N} \to \{C_1, C_2, \ldots, C_K\}, \tag{3}$$

where C_k is a subset of the total variable set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ and its cardinality $|C_k|$ denotes the number of variables in this cluster. The objective of the clustering function $\mathcal{C}$ is defined as follows:

$$\operatorname{argmax}_{\{C_1, C_2, \ldots, C_K\}} \sum_{k=1}^{K} \sum_{\mathbf{x}_i, \mathbf{x}_j \in C_k} S\left(\mathbf{x}_i, \mathbf{x}_j\right), \tag{4}$$

where $S(\mathbf{x}_i, \mathbf{x}_j)$ represents the similarity between variables $\mathbf{x}_i$ and $\mathbf{x}_j$. Then we obtain the permutation matrix $P \in \mathbb{R}^{N \times N}$:

$$P_{ij} = \begin{cases} 1, & \text{if } j = \pi(i), \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

where π is the permutation vector that orders the variables according to the clusters. Finally, the reordered matrix $\mathbf{X}'$ with columns permuted according to π is given by:

$$\mathbf{X}' = \mathbf{X} \cdot P. \tag{6}$$

In this reordered matrix $\mathbf{X}'$, rows (i.e., variables) are grouped according to the clusters.

3.4 Multi-scale Feature Extraction

To further capture features from different scales for fine-grained imputation, the reordered data $\mathbf{X}' \in \mathbb{R}^{N \times T}$ is fed to multiple convolutional layers with varying scales. Each convolutional layer l refers to kernel size k_l, stride s_l, padding p_l, and the number of output channels c_l. For each output channel i in the l^{th} convolutional layer, the convolution operation can be formulated as:

$$\Phi_l\left(\mathbf{X}'\right)_{i,t} = \sum_{j=1}^{k_l} W_{l,i,j} \cdot \mathbf{X}'_{t+j-p_l} + b_{l,i}, \tag{7}$$

where W and b denote the weight matrix and bias matrix respectively. Then these features of different scales $\Phi_l\left(\mathbf{X}'\right) \in \mathbb{R}^{c_l \times (T - k_l + 2p_l + 1)}$ are concatenated to obtained the multi-scale features $\dot{\mathbf{X}} \in \mathbb{R}^{\sum_{l=1}^{L} c_l \times \left(T' - k_l + 2p_l + 1\right)}$. Finally, these features are fed to the transformer encoder together with the initialized INR tokens to predict the INR tokens.

3.5 INR Continuous Function

INR continuous function f maps the timestamp t to time series data:

$$f : t \in \mathbb{R} \mapsto \mathbf{X}(t) \in \mathbb{R}^N, \tag{8}$$

where $\mathbf{X}(t)$ represents the output values of N variables at timestamp t. To effectively capture the complicated temporal patterns and successfully model the multiple variables, we design a novel form of INR continuous function. Following the previous work [Fons *et al.*, 2022; Li *et al.*, 2024], our INR continuous function includes three components to model trend, seasonal, and residual patterns separately. It can be defined as follows:

$$\hat{\mathbf{X}}(t) = f(t) = f_{tre}(t) + f_{sea}(t) + f_{res}(t), \tag{9}$$

where t is the input timestamp and $f(t)$ denotes the output (i.e., imputed data). The parameters in the INR continuous function are predicted by the transformer encoder (i.e., INR tokens).

Trend and Seasonal

The trend represents the long-term movement or direction of the time series data. It is typically smooth and reflects gradual shifts in the level of the time series, free from noise or short-term fluctuations. Mathematically, it can be modeled as a polynomial function:

$$f_{tre}(t) = \sum_{i=0}^{m} \alpha_i t^i, \tag{10}$$

where α_i denotes the coefficients and m refers to the degree of the polynomial. In addition, the seasonal component focuses on the repeating patterns or cycles in the time series data, representing predictable fluctuations due to seasonality or recurring events. These regular, cyclical, and short-term fluctuations are modeled with a periodic function:

$$f_{sea}(t) = \sum_{i=1}^{\lfloor T/2-1 \rfloor} \left(\beta_i \sin\left(2\pi i t\right) + \gamma_{i+\lfloor T/2 \rfloor} \cos\left(2\pi i t\right)\right), \tag{11}$$

where β_i and γ_i are Fourier coefficients.

Residual: Adaptive Group-based Architecture

The residual component represents the unexplained variation after removing the trend and seasonal effects, often modeled as a stochastic process. It is challenging to capture this complex information. As discussed in Figure 3, we find that regardless of the order of the variables, using a single MLP is not effective in modeling multiple variables from different distributions. However, if variables from the same distribution are represented using the same set of MLP layers, the performance will significantly improve. We define such a set as a group. In addition, the layers in the MLP that extract information across all variables are called global layers, while the layers within groups are referred to as group layers. The number of groups and their outputs are determined by the results of variable clustering, which allows our architecture to adapt to datasets with various characteristics. It is worth noting that when variables with different distributions are in the same group, the representation capability is significantly reduced. This proves the importance of the correlation information between the variables.

Theoretically, for any given timestamp t, we design L_1 global layers, L_2 group layers, and K groups. K is determined by the results of variable clustering. The global layers are given as follows:

$$h^{(0)} = t, \tag{12}$$

$$h^{(l_1)} = \sigma\left(W^{(l_1)} h^{(l_1-1)} + b^{(l_1)}\right), \tag{13}$$

where $l_1 \in [1, L_1]$, $h^{(l_1)}$ is the output of the l_1^{th} global layer, W and b are weight matrix and bias matrix. Then, for group g_k, the input is the output of the last global layer:

$$\hat{x}_{g_k}^{(0)} = h^{(L_1)}, \tag{14}$$

$$\hat{x}_{g_k}^{(l_2)} = \sigma\left(W_{g_k}^{(l_2)} \hat{x}_{g_k}^{(l_2-1)} + b_{g_k}^{(l_2)}\right), \tag{15}$$

where $l_2 \in [1, L_2]$, $\hat{x}_{g_k}^{l_2}$ refers to the output of the l_2^{th} group layer in group g_k, W and b are weight matrix and bias matrix. $\hat{x}_{g_k}^{L_2} \in \mathbb{R}^{|C_k|}$ and $|C_k|$ is the number of variables in the k^{th} cluster. The final output is the concatenation of the outputs from the last group layer of each group:

$$f_{res}(t) = \hat{x}_{g_1}^{(L_2)} \oplus \hat{x}_{g_2}^{(L_2)} \oplus \ldots \oplus \hat{x}_{g_K}^{(L_2)}. \tag{16}$$

4 Experiments

4.1 Experimental Setup

Datasets and Baseline Methods

We use eight time series imputation benchmark datasets to validate the performance of ImputeINR, including ETT [Zhou *et al.*, 2021], Weather [Institute, 2020], BAQ [Zhang *et al.*, 2017], IAQ [Repository, 2008], Solar [Laboratory, 2006], Phy2012 [Silva *et al.*, 2012], Phy2019 [Reyna *et al.*, 2019]), and MIMIC3 [Johnson *et al.*, 2018]. Further, we select the Phy2012, Phy2019, and MIMIC3 healthcare datasets to validate the effectiveness of the imputation results produced by ImputeINR for downstream disease diagnosis tasks. To verify the superiority of ImputeINR, we compare our method to nine state-of-the-art imputation methods, including RNN-based methods (BRITS [Cao *et al.*, 2018]), CNN-based methods (TimesNet [Wu *et al.*, 2023]), MLP-based methods (TimeMixer [Wang *et al.*, 2024]), transformer-based methods (SAITS [Du *et al.*, 2023], FPT [Zhou *et al.*, 2023], iTransformer [Liu *et al.*, 2024], ImputeFormer [Nie *et al.*, 2024]), and diffusion-based methods (CSDI [Tashiro *et al.*, 2021], SSSD [Alcaraz and Strodthoff, 2022]).

Experimental Settings

We apply the same data processing techniques and parameter settings. A sliding window approach is used, with a fixed window size of 48 for the Phy2012, Phy2019, and MIMIC3 datasets, and 96 for all other datasets. These settings follow those used in previous work [Wu *et al.*, 2023; Du, 2023]. To evaluate the imputation performance, we use the same masking strategy as previous works [Wu *et al.*, 2023; Zhou *et al.*, 2023], which randomly mask values in $\mathbf{X}_{\text{gt}}$ based on the mask rate r. For the imputation results, the multi-scale feature extraction module uses three parallel convolutional layers with kernel sizes of 3,5,7 respectively. The adaptive group-based architecture in the INR continuous function involves one global layer and one group layer within the residual component, with hidden dimensions set to 16. The transformer encoder consists of 6 blocks. We use the agglomerative clustering method to achieve variable clustering since it

Methods		ImputeINR		ImputeFormer		TimeMixer		iTransformer		FPT		TimesNet		SAITS		BRITS		SSSD		CSDI	
	Mask Rate	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETT	10%	0.020	0.098	0.021	0.091	0.035	0.115	0.042	0.141	**0.017**	**0.087**	0.018	0.088	0.021	0.100	0.021	0.089	0.022	0.098	0.019	**0.087**
	30%	0.027	0.109	**0.023**	0.098	0.041	0.125	0.066	0.180	0.030	0.110	0.031	0.111	0.030	0.114	0.028	0.110	0.027	0.109	0.024	**0.097**
	50%	**0.028**	**0.111**	0.034	0.116	0.054	0.143	0.109	0.234	0.041	0.130	0.035	0.123	0.031	0.116	0.040	0.130	0.029	0.112	0.034	0.112
	70%	**0.039**	**0.134**	0.050	0.142	0.077	0.170	0.124	0.246	0.085	0.181	0.057	0.155	0.043	0.135	0.068	0.181	0.044	0.135	0.049	0.135
	90%	**0.095**	**0.214**	0.122	0.218	0.223	0.276	0.247	0.336	0.272	0.309	0.231	0.295	0.213	0.218	0.251	0.358	0.154	0.254	0.124	0.216
Weather	10%	**0.026**	**0.063**	0.032	0.076	0.029	0.069	0.036	0.081	0.028	0.064	0.028	0.064	0.031	0.073	0.027	**0.063**	0.036	0.069	0.039	0.065
	30%	**0.030**	**0.072**	0.033	0.080	0.032	0.080	0.051	0.113	0.035	0.075	0.031	0.073	0.035	0.077	0.031	0.073	0.039	0.073	0.043	0.074
	50%	**0.031**	**0.073**	0.037	0.084	0.037	0.076	0.069	0.144	0.043	0.076	0.036	0.076	0.041	0.091	0.035	0.077	0.040	0.075	0.048	0.076
	70%	**0.036**	**0.082**	0.074	0.097	0.045	0.086	0.078	0.147	0.053	0.087	0.043	0.084	0.047	0.096	0.042	0.085	0.053	0.086	0.057	0.093
	90%	**0.065**	**0.123**	0.082	0.126	0.076	0.126	0.124	0.191	0.089	0.129	0.073	0.125	0.066	0.124	0.090	0.130	0.089	0.128	0.088	0.127
BAQ	10%	**0.083**	**0.169**	1.050	0.747	0.165	0.172	0.235	0.258	0.215	0.224	0.262	0.266	1.085	0.748	0.208	0.175	0.306	0.357	0.201	0.171
	30%	**0.096**	**0.171**	1.096	0.749	0.205	0.193	0.308	0.321	0.231	0.229	0.292	0.267	1.088	0.749	0.210	0.186	0.368	0.365	0.203	0.174
	50%	**0.101**	**0.172**	1.106	0.750	0.274	0.237	0.404	0.399	0.285	0.242	0.318	0.269	1.112	0.750	0.211	0.191	0.392	0.401	0.209	0.183
	70%	**0.117**	**0.181**	1.119	0.751	0.359	0.289	0.556	0.488	0.325	0.262	0.341	0.280	1.124	0.751	0.230	0.206	0.462	0.424	0.229	0.192
	90%	**0.122**	**0.185**	1.129	0.752	0.503	0.367	0.803	0.615	0.430	0.301	0.427	0.317	1.127	0.752	0.411	0.308	0.671	0.532	0.322	0.218
IAQ	10%	**0.007**	**0.061**	1.340	0.725	0.139	0.171	0.592	0.466	0.228	0.264	0.248	0.286	1.277	0.735	0.164	0.210	0.144	0.185	0.064	0.116
	30%	**0.008**	**0.062**	1.377	0.738	0.244	0.242	0.639	0.503	0.237	0.271	0.262	0.290	1.442	0.755	0.224	0.243	0.199	0.192	0.074	0.122
	50%	**0.009**	**0.063**	1.424	0.753	0.375	0.306	0.783	0.556	0.291	0.305	0.274	0.297	1.461	0.757	0.241	0.273	0.213	0.216	0.101	0.144
	70%	**0.010**	**0.068**	1.466	0.757	0.527	0.377	0.907	0.618	0.426	0.357	0.304	0.314	1.472	0.761	0.504	0.355	0.343	0.280	0.238	0.225
	90%	**0.029**	**0.116**	1.478	0.761	0.847	0.498	1.205	0.767	0.811	0.504	0.720	0.477	1.493	0.764	0.981	0.505	0.941	0.523	0.657	0.428
Solar	10%	**0.022**	**0.074**	0.768	0.771	0.024	0.079	0.060	0.167	0.075	0.173	0.048	0.132	0.770	0.772	0.023	0.075	0.122	0.143	0.023	0.075
	30%	**0.023**	**0.075**	0.770	0.772	0.034	0.107	0.071	0.181	0.084	0.185	0.049	0.133	0.771	0.773	0.024	0.076	0.128	0.147	0.025	0.078
	50%	**0.024**	**0.078**	0.772	0.773	0.052	0.143	0.079	0.189	0.101	0.202	0.052	0.139	0.772	0.774	0.026	0.080	0.145	0.159	0.026	0.080
	70%	**0.025**	**0.079**	0.773	0.774	0.075	0.173	0.088	0.200	0.139	0.243	0.061	0.151	0.773	0.775	0.030	0.085	0.211	0.228	0.036	0.084
	90%	**0.026**	**0.081**	0.774	0.775	0.166	0.249	0.120	0.250	0.435	0.444	0.121	0.211	0.774	0.776	0.052	0.100	0.426	0.312	0.041	0.085
Phy2012	10%	**0.072**	**0.096**	0.200	0.153	0.104	0.115	0.097	0.108	0.087	0.104	0.080	0.101	0.200	0.163	0.097	0.100	0.080	0.098	0.656	0.514
	30%	**0.079**	**0.101**	0.205	0.155	0.117	0.120	0.099	0.111	0.099	0.111	0.103	0.108	0.203	0.168	0.108	0.105	0.107	0.102	0.828	0.585
	50%	**0.092**	**0.107**	0.210	0.158	0.142	0.124	0.109	0.115	0.105	0.118	0.145	0.118	0.208	0.173	0.117	0.116	0.139	0.108	0.923	0.601
	70%	**0.071**	**0.112**	0.229	0.169	0.148	0.129	0.124	0.120	0.132	0.131	0.149	0.128	0.237	0.195	0.125	0.123	0.149	0.118	0.994	0.670
	90%	**0.127**	**0.124**	0.232	0.170	0.179	0.143	0.160	0.135	0.167	0.145	0.177	0.144	0.214	0.159	0.163	0.139	0.226	0.157	1.068	0.705
Phy2019	10%	**0.071**	**0.102**	0.199	0.159	0.100	0.116	0.072	0.104	0.082	0.111	0.075	0.105	0.199	0.168	0.089	0.103	0.083	0.109	0.718	0.579
	30%	**0.079**	**0.109**	0.206	0.160	0.104	0.120	0.098	0.122	0.091	0.116	0.084	0.111	0.203	0.169	0.099	0.110	0.085	0.114	0.813	0.648
	50%	**0.087**	**0.115**	0.209	0.164	0.109	0.125	0.100	0.123	0.102	0.124	0.094	0.118	0.204	0.175	0.109	0.118	0.098	0.118	0.956	0.657
	70%	**0.098**	**0.120**	0.211	0.172	0.119	0.132	0.112	0.129	0.116	0.133	0.109	0.128	0.205	0.178	0.122	0.124	0.167	0.142	0.985	0.673
	90%	**0.121**	**0.131**	0.214	0.174	0.152	0.149	0.123	0.132	0.153	0.152	0.149	0.149	0.206	0.180	0.151	0.142	0.226	0.173	1.061	0.761
MIMIC3	10%	**0.019**	**0.041**	0.149	0.142	0.049	0.077	0.052	0.085	0.031	0.055	0.043	0.070	0.150	0.148	0.031	0.045	0.036	0.045	0.096	0.067
	30%	**0.023**	**0.044**	0.150	0.143	0.081	0.079	0.055	0.089	0.033	0.057	0.051	0.072	0.151	0.149	0.039	0.046	0.094	0.050	0.157	0.084
	50%	**0.027**	**0.048**	0.151	0.144	0.075	0.082	0.061	0.100	0.040	0.063	0.076	0.074	0.152	0.150	0.066	0.057	0.102	0.058	0.216	0.088
	70%	**0.033**	**0.056**	0.161	0.145	0.083	0.086	0.084	0.113	0.081	0.068	0.093	0.080	0.155	0.153	0.083	0.059	0.110	0.064	0.220	0.114
	90%	**0.062**	**0.070**	0.178	0.148	0.126	0.097	0.136	0.130	0.098	0.080	0.134	0.094	0.169	0.156	0.112	0.077	0.128	0.078	0.354	0.201
Average		**0.054**	**0.102**	0.496	0.371	0.158	0.164	0.232	0.238	0.161	0.176	0.148	0.166	0.499	0.376	0.142	0.146	0.186	0.178	0.325	0.260

Table 1: Imputation results. The best results are in **Bold**. And the second ones are underlined.

adopts diverse inputs without the need to pre-specify the number of clusters. For the downstream disease diagnosis task, we employ the default Long Short-Term Memory (LSTM) classifier provided by the official benchmark [Harutyunyan *et al.*, 2019] as the disease diagnosis model.Experiments are performed using the ADAM optimizer [Kingma, 2014] with an initial learning rate of 10^{-3}. All experiments are conducted on a single 24GB GeForce RTX 3090 GPU.

4.2 Imputation Task

Main Results

We compare our ImputeINR method to nine state-of-the-art imputation methods with five different mask rates r. As shown in Table 1, our ImputeINR achieves the best performance in most conditions in terms of both MSE and MAE metrics. Overall, across all datasets and mask rates, our method achieves an average MSE reduction of 62.0% compared to the second-best results. These results demonstrate that our proposed ImputeINR can effectively deal with datasets of various sizes. In addition, we observe that the performance of most methods declines as the mask rate r increases. This aligns with our expectations, as fewer samples are captured leading to incomplete information, which increases the difficulty of imputation. However, ImputeINR is still effective even with an extreme mask rate. When 90% of the data is masked, the average MSE of our method is reduced by 68.2% compared to the second-best ones. This indicates that ImputeINR can learn continuous functions from very few data points, achieving fine-grained imputation.

Ablation Studies

In this section, we conduct ablation studies to evaluate the effectiveness of the multi-scale feature extraction block, variable clustering, and adaptive group-based architecture. We set mask rate r to be 50% for the ablation study, as it provides

Multi-scale Features	Variable Clustering	Adaptive Group	ETT MSE	ETT MAE	Weather MSE	Weather MAE	BAQ MSE	BAQ MAE	IAQ MSE	IAQ MAE	Solar MSE	Solar MAE	Phy2012 MSE	Phy2012 MAE	Phy2019 MSE	Phy2019 MAE	MIMIC3 MSE	MIMIC3 MAE
✗	✗	✗	0.039	0.135	0.038	0.083	0.227	0.262	0.018	0.092	0.036	0.106	0.099	0.114	0.098	0.119	0.038	0.062
✗	✗	✓	0.036	0.130	0.035	0.081	0.222	0.258	0.015	0.084	0.034	0.098	0.099	0.113	0.096	0.117	0.036	0.056
✗	✓	✗	0.036	0.129	0.036	0.080	0.218	0.259	0.015	0.083	0.033	0.096	0.098	0.113	0.097	0.118	0.035	0.057
✓	✗	✗	0.035	0.127	0.035	0.082	0.209	0.252	0.017	0.088	0.033	0.100	0.097	0.114	0.095	0.117	0.037	0.055
✗	✓	✓	0.029	0.115	0.032	0.074	0.192	0.243	0.010	0.066	0.031	0.092	0.093	0.108	0.088	0.111	0.028	0.049
✓	✗	✓	0.034	0.124	0.034	0.079	0.203	0.248	0.012	0.077	0.031	0.094	0.095	0.113	0.093	0.116	0.033	0.053
✓	✓	✗	0.033	0.123	0.033	0.078	0.199	0.244	0.014	0.081	0.032	0.096	0.096	0.113	0.094	0.117	0.035	0.051
✓	✓	✓	**0.028**	**0.111**	**0.031**	**0.073**	**0.101**	**0.172**	**0.009**	**0.063**	**0.024**	**0.078**	**0.092**	**0.107**	**0.087**	**0.115**	**0.027**	**0.048**

Table 2: The ablation studies of each module in ImputeINR. The best results are in **Bold**.

Dataset	ImputeINR	ImputeFormer	TimeMixer	iTransformer	FPT	TimesNet	SAITS	BRITS	SSSD	CSDI	Mean	Zero
Phy2012	**0.8382**	0.8238	0.8304	0.8305	0.8230	0.8286	0.8299	0.8289	0.8215	0.8325	0.8175	0.8023
Phy2019	**0.7346**	0.7050	0.6923	0.6737	0.7117	0.6964	0.6743	0.6968	0.6818	0.6980	0.7095	0.6871
MIMIC3	**0.8604**	0.8482	0.8570	0.8564	0.8540	0.8533	0.8500	0.8527	0.8096	0.8526	0.8472	0.8419

Table 3: The disease diagnosis results. The AUROC values are reported and the best results are in **Bold**.

a moderate level of missingness that effectively highlights the impact of each model component while maintaining stability and representativeness in the results. Table 2 presents the imputation results for all conditions. First, the model without any of the three modules exhibits the lowest performance. Building on this, adding any one of the modules will enhance the imputation capability of the model. This individually validates the effectiveness of each of the three modules. Furthermore, the permutation of any two modules will lead to higher performance. Among them, the combination of variable clustering and adaptive group-based architecture yields the best results. This is as expected, since the variable clustering determines the variable partition and its outcomes correspond directly to the number of groups. Therefore, these two modules can support each other, facilitating better representational learning. Finally, the model using all three modules displays the highest imputation performance as each module contributes complementary strengths.

4.3 Downstream Disease Diagnosis Task

To evaluate the effectiveness of our proposed ImputeINR method in real-world healthcare applications, we conduct disease diagnosis using the imputed data. Specifically, we first apply ImputeINR to impute missing values in healthcare datasets and then use the imputed data for disease diagnosis. As shown in Table 3, the performance is compared against commonly used imputation methods, including zero imputation, mean imputation, and other state-of-the-art imputation techniques. For evaluation, we employ the standard metric of area under the receiver operating characteristic curve (AUROC). The results demonstrate that the imputed data from ImputeINR consistently leads to superior diagnostic performance across all metrics. This indicates that our model not only effectively reconstructs missing values but also preserves critical disease-related patterns, enhancing the downstream diagnostic capability.

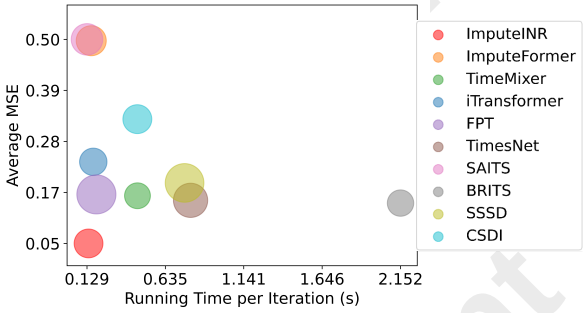

Figure 4: A bubble chart reporting running time vs. imputation performance. The size of each bubble refers to the model size.

4.4 Efficiency Analysis

We evaluate the efficiency of the ImputeINR method, with the results presented in Figure 4. On the x-axis, we plot the running time per iteration, while the y-axis represents the average MSE. The size of each bubble is proportional to the number of model parameters.This figure shows that ImputeINR achieves accurate imputation results while requiring relatively little running time. Moreover, the corresponding bubble size suggests that ImputeINR has a relatively small number of model parameters, further emphasizing its efficiency. These results prove that ImputeINR strikes a favorable balance between computational efficiency and imputation accuracy.

5 Conclusion

In this paper, we propose ImputeINR, which achieves effective imputation in extremely sparse scenarios and the imputed data can further enhance the performance of downstream disease diagnosis tasks. In contrast to existing imputation approaches, ImputeINR leverages the sampling frequency-independent and infinite-sampling frequency capabilities of INR to achieve fine-grained imputation with sparse data.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant Nos. 62202422 and 62372408).

References

[Afrifa-Yamoah *et al.*, 2020] Eben Afrifa-Yamoah, Ute A Mueller, Stephen M Taylor, and Aiden J Fisher. Missing data imputation of high-resolution temporal climate time series data. *Meteorological Applications*, 27(1):e1873, 2020.

[Alcaraz and Strodthoff, 2022] Juan Miguel Lopez Alcaraz and Nils Strodthoff. Diffusion-based time series imputation and forecasting with structured state space models. *arXiv preprint arXiv:2208.09399*, 2022.

[Altman, 1992] Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[Cao *et al.*, 2018] 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.

[Chen and Wang, 2022] Yinbo Chen and Xiaolong Wang. Transformers as meta-learners for implicit neural representations. In *European Conference on Computer Vision*, pages 170–187. Springer, 2022.

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

[Du, 2023] Wenjie Du. PyPOTS: a Python toolbox for data mining on Partially-Observed Time Series. *arXiv preprint arXiv:2305.18811*, 2023.

[Fons *et al.*, 2022] Elizabeth Fons, Alejandro Sztrajman, Yousef El-Laham, Alexandros Iosifidis, and Svitlana Vyetrenko. Hypertime: Implicit neural representation for time series. *arXiv preprint arXiv:2208.05836*, 2022.

[Fung, 2006] David S Fung. Methods for the estimation of missing values in time series. 2006.

[Guo *et al.*, 2025] Jialong Guo, Ke Liu, Jiangchao Yao, Zhihua Wang, Jiajun Bu, Haishuai Wang, et al. Metanerv: Meta neural representations for videos with spatial-temporal guidance. *arXiv preprint arXiv:2501.02427*, 2025.

[Harutyunyan *et al.*, 2019] Hrayr Harutyunyan, Hrant Khachatrian, David C. Kale, Greg Ver Steeg, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific Data*, 6(1):96, 2019.

[Institute, 2020] Max Planck Institute. Bgc jena: Weather station data. https://www.bgc-jena.mpg.de/wetter/, 2020. Accessed: 2025-05-27.

[Jeong and Shin, 2022] Kyeong-Joong Jeong and Yong-Min Shin. Time-series anomaly detection with implicit neural representation. *arXiv preprint arXiv:2201.11950*, 2022.

[Johnson *et al.*, 2018] Alistair E W Johnson, David J Stone, Leo A Celi, and Tom J Pollard. The mimic code repository: enabling reproducibility in critical care research. *Journal of the American Medical Informatics Association*, 25(1):32–39, 2018.

[Kingma, 2014] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Laboratory, 2006] National Renewable Energy Laboratory. Solar power data for integration studies. https://www.nrel.gov/grid/solar-power-data.html, 2006. Accessed: 2025-05-27.

[Lee *et al.*, 2021] Jaeho Lee, Jihoon Tack, Namhoon Lee, and Jinwoo Shin. Meta-learning sparse implicit neural representations. *Advances in Neural Information Processing Systems*, 34:11769–11780, 2021.

[Li *et al.*, 2023a] Mengxuan Li, Peng Peng, Haiyue Sun, Min Wang, and Hongwei Wang. An order-invariant and interpretable dilated convolution neural network for chemical process fault detection and diagnosis. *IEEE Transactions on Automation Science and Engineering*, 2023.

[Li *et al.*, 2023b] Mengxuan Li, Peng Peng, Jingxin Zhang, Hongwei Wang, and Weiming Shen. Sccam: Supervised contrastive convolutional attention mechanism for ante-hoc interpretable fault diagnosis with limited fault samples. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[Li *et al.*, 2024] Mengxuan Li, Ke Liu, Hongyang Chen, Jiajun Bu, Hongwei Wang, and Haishuai Wang. Tsinr: Capturing temporal continuity via implicit neural representations for time series anomaly detection. *arXiv preprint arXiv:2411.11641*, 2024.

[Liu *et al.*, 2023a] Ke Liu, Feng Liu, Haishuai Wang, Ning Ma, Jiajun Bu, and Bo Han. Partition speeds up learning implicit neural representations based on exponential-increase hypothesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5474–5483, 2023.

[Liu *et al.*, 2023b] Ke Liu, Ning Ma, Zhihua Wang, Jingjun Gu, Jiajun Bu, and Haishuai Wang. Implicit neural distance optimization for mesh neural subdivision. In *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pages 2039–2044. IEEE, 2023.

[Liu *et al.*, 2024] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024.

[Mai and Liu, 2022] Long Mai and Feng Liu. Motion-adjustable neural implicit video representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10738–10747, 2022.

[Molaei *et al.*, 2023] Amirali Molaei, Amirhossein Aminimehr, Armin Tavakoli, Amirhossein Kazerouni, Bobby

Azad, Reza Azad, and Dorit Merhof. Implicit neural representation in medical imaging: A comparative survey. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2381–2391, 2023.

[Naour *et al.*, 2023] Etienne Le Naour, Louis Serrano, Léon Migus, Yuan Yin, Ghislain Agoua, Nicolas Baskiotis, Patrick Gallinari, and Vincent Guigue. Time series continuous modeling for imputation and forecasting with implicit neural representations. *arXiv preprint arXiv:2306.05880*, 2023.

[Nie *et al.*, 2024] 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*, pages 2260–2271, 2024.

[Repository, 2008] UCI Machine Learning Repository. Air quality data set. https://archive.ics.uci.edu/dataset/360/air+quality, 2008. Accessed: 2025-05-27.

[Reyna *et al.*, 2019] MA Reyna, C Josef, R Jeter, SP Shashikumar, MB Westover, S Nemati, GD Clifford, and A Sharma. Early prediction of sepsis from clinical data: the physionetcomputing in cardiology challenge 2019. *Critical Care Medicine*, 48(2):210–217, 2019.

[Saragadam *et al.*, 2022] Vishwanath Saragadam, Jasper Tan, Guha Balakrishnan, Richard G Baraniuk, and Ashok Veeraraghavan. Miner: Multiscale implicit neural representation. In *European Conference on Computer Vision*, pages 318–333. Springer, 2022.

[Schaffer *et al.*, 2021] Andrea L Schaffer, Timothy A Dobbins, and Sallie-Anne Pearson. Interrupted time series analysis using autoregressive integrated moving average (arima) models: a guide for evaluating large-scale health interventions. *BMC medical research methodology*, 21:1–12, 2021.

[Shan *et al.*, 2023] 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)*, pages 1–5. IEEE, 2023.

[Silva *et al.*, 2012] 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. *Computing in cardiology*, 39:245, 2012.

[Tashiro *et al.*, 2021] 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.

[Van Buuren and Groothuis-Oudshoorn, 2011] Stef Van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, 45:1–67, 2011.

[Wang *et al.*, 2018] Haishuai Wang, Jia Wu, Peng Zhang, and Yixin Chen. Learning shapelet patterns from network-based time series. *IEEE transactions on industrial informatics*, 15(7):3864–3876, 2018.

[Wang *et al.*, 2024] Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and JUN ZHOU. Timemixer: Decomposable multiscale mixing for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024.

[Wu *et al.*, 2023] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations*, 2023.

[Yang *et al.*, 2024] Yiying Yang, Fukun Yin, Wen Liu, Jiayuan Fan, Xin Chen, Gang Yu, and Tao Chen. Pm-inr: Prior-rich multi-modal implicit large-scale scene neural representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 6594–6602, 2024.

[Yin *et al.*, 2022] Fukun Yin, Wen Liu, Zilong Huang, Pei Cheng, Tao Chen, and Gang Yu. Coordinates are not lonely-codebook prior helps implicit neural 3d representations. *Advances in Neural Information Processing Systems*, 35:12705–12717, 2022.

[Yoon *et al.*, 2018] 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.

[Zeng *et al.*, 2023] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128, 2023.

[Zhang *et al.*, 2017] Shuyi Zhang, Bin Guo, Anlan Dong, Jing He, Ziping Xu, and Song Xi Chen. Cautionary tales on air-quality improvement in beijing. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473, 2017.

[Zhang *et al.*, 2024] Shuyi Zhang, Ke Liu, Jingjun Gu, Xiaoxu Cai, Zhihua Wang, Jiajun Bu, and Haishuai Wang. Attention beats linear for fast implicit neural representation generation. In *European Conference on Computer Vision*, pages 1–18. Springer, 2024.

[Zhou *et al.*, 2021] 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, pages 11106–11115, 2021.

[Zhou *et al.*, 2023] Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36:43322–43355, 2023.