# SADI: Similarity-Aware Diffusion Model-Based Imputation for Incomplete Temporal EHR Data

**Zongyu Dai**
University of Pennsylvania
daizy@sas.upenn.edu

**Emily Getzen**
University of Pennsylvania
getzene@pennmedicine.upenn.edu

**Qi Long**
University of Pennsylvania
qlong@upenn.edu

## Abstract

Missing values are prevalent in temporal electronic health records (EHR) data and are known to complicate data analysis and lead to biased results. The current state-of-the-art (SOTA) models for imputing missing values in EHR primarily leverage correlations across time points and across features, which perform well when data have strong correlation across time points, such as in ICU data where high-frequency time series data are collected. However, this is often insufficient for temporal EHR data from non-ICU settings (e.g., outpatient visits for primary care or specialty care), where data are collected at substantially sparser time points, resulting in much weaker correlation across time points. To address this methodological gap, we propose the Similarity-Aware Diffusion Model-Based Imputation (SADI), a novel imputation method that leverages the diffusion model and utilizes information across dependent variables. We apply SADI to impute incomplete temporal EHR data and propose a similarity-aware denoising function, which includes a self-attention mechanism to model the correlations between time points, features, and similar patients. To the best of our knowledge, this is the first time that the information of similar patients is directly used to construct imputation for incomplete temporal EHR data. Our extensive experiments on two datasets, the Critical Path For Alzheimer's Disease (CPAD) data and the PhysioNet Challenge 2012 data, show that SADI outperforms the current SOTA under various missing data mechanisms, including missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR).

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238. Copyright 2024 by the author(s).

## 1 INTRODUCTION

Electronic health records (EHR) are comprehensive collections of patient health information that include demographic data, medical history, laboratory results, diagnoses, treatment, and more (Häyrinen et al., 2008). Rich data in EHR offer great promises in advancing research and improving patient care (Cowie et al., 2017; Jensen et al., 2012; Rajkomar et al., 2018b). For example, deep learning models trained on temporal EHR data can detect sepsis at an early stage (Khojandi et al., 2018; Lauritsen et al., 2020), potentially reducing mortality rates. However, EHR data are typically recorded with irregular time intervals and contain a significant amount of missing values(Wells et al., 2013; Steele et al., 2018), which present daunting challenges for many statistical and machine learning models that require structured, regularly sampled, and complete input data. More importantly, biases caused by missing data in EHR have been identified as a significant factor contributing to the unfairness of ML/AI models in medicine, which can perpetuate and exacerbate health inequities (Gianfrancesco et al., 2018; Rajkomar et al., 2018a). For example, patients with less access to healthcare, often people of color or with lower socioeconomic status, tend to have more incomplete data in their EHR (Getzen et al., 2023). Getzen et al. further demonstrate that incomplete data in EHR adversely impact the accuracy of prediction models which would unfairly harm under-served minority groups and exacerbate health inequities (Getzen et al., 2023). As such, it is of great value to develop robust methods for adequately addressing incomplete temporal EHR data.

Imputation is a widely used and effective approach for addressing the issue of missing data in EHR data. In recent years, there has been a growing body of literature on deep learning-based imputation models for time series EHR data (Cao et al., 2018; Fortuin et al., 2020; Tashiro et al., 2021). The current state-of-the-art (SOTA) models can generally be classified into two categories: RNN-based imputation models, such as those presented in Che et al. (2018); Yoon et al. (2018); Luo et al. (2018, 2019); Cao et al. (2018), and transformer-based imputation models, such as Suo et al. (2020); Shukla and Marlin (2021); Tashiro et al. (2021). For example, Multi-Directional Recurrent Neural Network (MRNN) (Yoon et al., 2018) leverages the power of a bi-directional recurrent neural network, and is composed of an interpolation block and an imputation block. It imputes missing values according to hidden states in both directions of RNN. Similar to MRNN, Bidirectional Recurrent Imputation for Time Series (BRITS) (Cao et al., 2018) also conducts imputation based on a bidirectional recurrent neural network. One difference is that BRITS treats missing values as variables in the model graph, and this change can lead to a more accurate estimation. However, all RNN-based imputation methods have an inherent weakness, which is RNN suffers from the short memory issue. Hence RNN-based imputation methods might not effectively model long-term dependencies. Transformer-based methods, which use a self-attention mechanism and are non-autoregressive, can generally overcome the short memory issue and lead to better imputation performance. For example, the Global and Local Time Series Imputation with Multi-directional Attention Learning (GLIMA) (Suo et al., 2020), which is a combination of RNN networks and transformer layers, imputes missing values by extracting local and global information from time series. The multi-Time Attention Networks for Irregularly Sampled Time Series (mTAND) (Shukla and Marlin, 2021) imputes missing values through an encoder-decoder framework, in which a newly designed attention mechanism is used to interpolate missing values. The Conditional Score-based Diffusion Model for Imputation (CSDI) (Tashiro et al., 2021) is a more recent SOTA imputation method that uses a 2D transformer to capture temporal and feature dependencies among EHR data.

It is important to note that all of the existing SOTA methods primarily leverage correlations across time points and across features to impute missing values. They use RNN or transformer layers to capture these dependencies. These models perform well when data have strong correlation across time points, such as in ICU data where high-frequency time series data are collected. However, this is often insufficient for temporal EHR data from non-ICU settings (e.g., outpatient visits for primary care or specialty care), where data are collected at substantially sparser time points, resulting in much weaker correlation across time points. In this case, it is crucial to also consider patient similarity in the imputation model. Intuitively, patients with similar characteristics and disease histories tend to have similar lab values, and the correlation between similar patients can also be leveraged for a more robust imputation model. In this paper, we propose a similarity-aware imputation model known as SADI (**S**imilarity-**A**ware **D**iffusion model-based **I**mputation for incomplete temporal EHR data) to impute missing values by modeling the dependencies across three dimensions: time, feature and patient. Particularly, our contributions are summarized as the following:

- We propose a similarity-aware diffusion model-based imputation method named SADI. Then we apply SADI to temporal EHR data and design a similarity-aware denoising function that models correlations from all three perspectives (time, feature, and patient). Thus, our imputation model can directly borrow information from similar patients. To the best of our knowledge, this is the first time people have modeled the patient dependency.

- We conduct extensive experiments to quantitatively evaluate our proposed approach under different missing mechanisms. Our experiments show that the SADI outperforms existing SOTA imputation models under MCAR, MAR, and MNAR, particularly for temporal EHRs data from non-ICU settings.

## 2 NOTATIONS

To fix ideas, let $\mathbf{X}_{i,t}^d$ ($i = 1, \ldots, n$; $t = 1, \ldots, T$; $d = 1, \ldots, p$) denote the observation for the $i$-th patient at time $t$ for the $d$-th feature. Here $n$ represents the number of patients (samples), $T$ represents the length of the temporal EHR data, and $p$ represents the number of features. Without loss of generality, we focus on the case that each patient's data has the same length $T$. For patients with fewer visits/time points, their EHR data will be extended through zero-padding, with these zeroes representing missing values. Denote the missing indicator of $\mathbf{X}_{i,t}^d$ as $\mathbf{M}_{i,t}^d$. If $\mathbf{M}_{i,t}^d = 1$, then $\mathbf{X}_{i,t}^d$ is observed. Similarly, if $\mathbf{M}_{i,t}^d = 0$, then $\mathbf{X}_{i,t}^d$ is missing. Additionally, $\mathbf{X}_i$ represents the full $p$-dimensional temporal EHR data for the $i$-th patient, and $\mathbf{X}_i^d = \left(\mathbf{X}_{i,1}^d, \ldots, \mathbf{X}_{i,T}^d\right)$ represents the $d$-th feature of $\mathbf{X}_i$, which is one-dimensional EHR data. Plus, all the following norms $\|\cdot\|$ represent $l_2$ norm and $[K]$ represents list of positive integers from 1 to $K$.

## 3 PRELIMINARIES

### 3.1 Adapting diffusion models for imputation

Diffusion models are a class of powerful generative models that have been used in many applications (Song and Ermon, 2019; Niu et al., 2020; Ho et al., 2020; Song et al., 2020; Kong et al., 2020; Chen et al., 2020). They can also be leveraged to approximate conditional distribution given observed data (Song et al., 2020; Kadkhodaie and Simoncelli, 2020; Mittal et al., 2021). More recently, diffusion models have also been adapted for imputing missing time series data by Tashiro et al. (2021), known as the Conditional Score-based Diffusion Model for Imputation (CSDI).

In this section, we briefly review the application of diffusion models for estimating the conditional distribution of missing data given the observed data. Given a sample $\mathbf{x}_0$ with missing values, which is not limited to time series, we are interested in two parts of $\mathbf{x}_0$: the target part, represented by $\mathbf{x}_0^{ta}$, and the conditional part, represented by $\mathbf{x}_0^{co}$. Our goal is to estimate the true conditional distribution $q(\mathbf{x}_0^{ta}|\mathbf{x}_0^{co})$ by the model distribution $p_\theta(\mathbf{x}_0^{ta}|\mathbf{x}_0^{co})$ from a diffusion process. It is important to note that, in section 3.1, the subscript $s$ of variable $\mathbf{x}$ does not represent a particular patient or time step in the time series. Instead, it represents a step in the Markov process. The first subscript 0 represents variables $\mathbf{x}_0$ coming from the true data distribution.

At a high level, the diffusion model estimates the true conditional distribution through two processes: a forward process and a reverse process. The forward process adds noise to the target part until it resembles a sample from a white noise Gaussian distribution. This is done through a Markov chain that generates a sequence of latent variables $\mathbf{x}_1^{ta}, \dots, \mathbf{x}_S^{ta}$ as follows:

$$q(\mathbf{x}_1^{ta}, \dots, \mathbf{x}_S^{ta}|\mathbf{x}_0^{ta}) = \prod_{s=1}^{S} q(\mathbf{x}_s^{ta}|\mathbf{x}_{s-1}^{ta})$$
$$\text{where} \qquad q(\mathbf{x}_s^{ta}|\mathbf{x}_{s-1}^{ta}) = \mathcal{N}\left(\sqrt{1-\beta_s}\mathbf{x}_{s-1}^{ta}, \beta_s\mathbf{I}\right) \tag{1}$$

Here, $\beta_s \in (0,1)$ are hyperparameters that represent the noise level. The marginal distribution for $\mathbf{x}_s^{ta}$ can be calculated as

$$q(\mathbf{x}_s^{ta}|\mathbf{x}_0^{ta}) = \mathcal{N}\left(\sqrt{\bar{\alpha}_s}\mathbf{x}_0^{ta}, (1-\bar{\alpha}_s)\mathbf{I}\right)$$

where $\alpha_s := 1 - \beta_s$ and $\bar{\alpha}_s := \prod_{i=1}^{s} \alpha_i$. The above equation is equivalent to

$$\mathbf{x}_s^{ta} = \sqrt{\bar{\alpha}_s}\mathbf{x}_0^{ta} + \sqrt{1-\bar{\alpha}_s}\boldsymbol{\epsilon} \tag{2}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ is a Gaussian vector.

The reverse process removes noise from $\mathbf{x}_S^{ta}$ to recover the original data $\mathbf{x}_0^{ta}$. This is done through a learnable Markov chain defined by the following distribution:

$$\begin{aligned} & p_\theta(\mathbf{x}_0^{ta}, \dots, \mathbf{x}_S^{ta}|\mathbf{x}_0^{co}) \\ & = p(\mathbf{x}_S^{ta}) \prod_{s=1}^{S} p_\theta(\mathbf{x}_{s-1}^{ta}|\mathbf{x}_s^{ta}, \mathbf{x}_0^{co}), \quad \text{where } \mathbf{x}_S^{ta} \sim \mathcal{N}(0, \mathbf{I}) \\ & p_\theta(\mathbf{x}_{s-1}^{ta}|\mathbf{x}_s^{ta}, \mathbf{x}_0^{co}) \\ & = \mathcal{N}(\mathbf{x}_{s-1}^{ta}; \boldsymbol{\mu}_\theta(\mathbf{x}_s^{ta}, s|\mathbf{x}_0^{co}), \sigma_\theta(\mathbf{x}_s^{ta}, s|\mathbf{x}_0^{co})\mathbf{I}) \end{aligned} \tag{3}$$

where $\theta$ represents model parameters. Note that all the terms are conditioned on $\mathbf{x}_0^{co}$ to exploit conditional observations. The conditional version of the Denoising Diffusion Probabilistic Model (DDPM) (Ho et al., 2020) is used in this method, which uses the following specific parameterization of $p_\theta(\mathbf{x}_{s-1}^{ta}|\mathbf{x}_s^{ta}, \mathbf{x}_0^{co})$:

$$\begin{aligned} \boldsymbol{\mu}_\theta(\mathbf{x}_s^{ta}, s|\mathbf{x}_0^{co}) &= \frac{1}{\alpha_s}\left(\mathbf{x}_s^{ta} - \frac{\beta_s}{\sqrt{1-\alpha_s}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_s^{ta}, s|\mathbf{x}_0^{co})\right) \\ \sigma_\theta(\mathbf{x}_s^{ta}, s|\mathbf{x}_0^{co}) &= \begin{cases} \sqrt{\frac{1-\alpha_{s-1}}{1-\alpha_s}\beta_s} & s > 1 \\ \sqrt{\beta_1} & s = 1 \end{cases} \end{aligned} \tag{4}$$

Here, $\boldsymbol{\epsilon}_\theta$ is a deep neural network with parameters $\theta$. Under this parameterization, the training of the reverse process is equivalent to solving the following optimization problem:

$$\begin{aligned} & \min_\theta \mathcal{L}(\theta) \\ & := \min_\theta \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), \boldsymbol{\epsilon} \sim \mathcal{N}(0,\mathbf{I}), s \sim \mathcal{U}\{1,S\}} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_s^{ta}, s|\mathbf{x}_0^{co})\|_2^2 \end{aligned} \tag{5}$$

where $\mathbf{x}_s^{ta} = \sqrt{\bar{\alpha}_s}\mathbf{x}_0^{ta} + \sqrt{1-\bar{\alpha}_s}\boldsymbol{\epsilon}$. This can be interpreted as using the function $\boldsymbol{\epsilon}_\theta$ as a denoising function to estimate the noise added to its noisy input $\mathbf{x}_s^{ta}$. Once the model is trained, we can sample $\mathbf{x}_0^{ta}$ from the reverse process eq. (3). The model distribution $p_\theta(\mathbf{x}_0^{ta}|\mathbf{x}_0^{co})$ is then used to estimate the true conditional distribution $q(\mathbf{x}_0^{ta}|\mathbf{x}_0^{co})$.

Since we have no access to the ground truth of missing values, both $\mathbf{x}_0^{ta}$ and $\mathbf{x}_0^{co}$ are selected from observed values in the model training phase (Tashiro et al., 2021). During the imputation phase, $\mathbf{x}_0^{ta}$ consists of all missing values and $\mathbf{x}_0^{co}$ consists of all observed values, in order to make full use of observed information.

### 3.2 Time series clustering

To gain insights on patient similarities, it is common to group patients' data into clusters. Numerous techniques have been proposed for clustering time series data, including the K-Means with Dynamic Time Warping (DTW) (Berndt and Clifford, 1994; Salvador and Chan, 2007), K-Shape algorithm (Paparrizos and

**Algorithm 1** Training of SADI

**Input:** distribution $q_k(\mathbf{x}_0)$ and the corresponding weight $c_k$ for $k \in [K]$, target selection strategy, training epochs $N$, maximum number of diffusion steps $S$, noise level $\{\beta_s\}_{s=1}^S$

**Output:** Trained denoising function $\boldsymbol{\epsilon}_\theta$

```
1: for $i \in \{1, \dots, N\}$ do
2:     Sample $k \sim \mathcal{C}\{c_1, c_2, \dots, c_K\}$
3:     Sample $s \sim \text{Unif}\{1, \dots, S\}$, and $\mathbf{x}_0 \sim q_k(\mathbf{x}_0)$
4:     Separate observed values $\mathbf{x}_0$ into target part $\mathbf{x}_0^{ta}$ and conditional part $\mathbf{x}_0^{co}$ by the target selection strategy
5:     Sample $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ where the dimension of $\boldsymbol{\epsilon}$ corresponds to $\mathbf{x}_0^{ta}$
6:     Calculate noisy targets $\mathbf{x}_s^{ta} = \sqrt{\bar{\alpha}_s}\mathbf{x}_0^{ta} + \sqrt{1-\bar{\alpha}_s}\boldsymbol{\epsilon}$
7:     Take gradient step on $\nabla \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_s^{ta}, s|\mathbf{x}_0^{co})\|_2^2$
8: end for
```

**Algorithm 2** Sampling/Imputation with SADI

**Input:** a data sample $\mathbf{x}_0$ from $q_k(\mathbf{x}_0)$, maximum number of diffusion steps $S$, trained denoising function $\boldsymbol{\epsilon}_\theta$

**Output:** Imputed values

```
1: Denote the observed values of $\mathbf{x}_0$ as $\mathbf{x}_0^{co}$, the missing part as $\mathbf{x}_0^{ta}$
2: Sample $\mathbf{x}_S^{ta} \sim \mathcal{N}(0, \mathbf{I})$ where $\mathbf{x}_S^{ta}$ has the same shape as $\mathbf{x}_0^{ta}$.
3: for $s \in \{S, \dots, 1\}$ do
4:     Sample $\mathbf{x}_{s-1}^{ta}$ from the reverse process, see eq. (3).
5: end for
```

Gravano, 2015), and K-Spectral Centroid algorithm (K-SC) (Yang and Leskovec, 2011; Ozer et al., 2020). The common idea behind these methods is to utilize an appropriate distance metric for time series data to define similarity between patients. All of them operate in an iterative manner, similar to the K-Means algorithm, and involve two steps: the assignment step and the refinement step. During the assignment step, each EHR time series is assigned to a centroid, with each centroid representing a cluster. The refinement step involves updating the centroids using the EHR data in their corresponding clusters. This process is repeated until the clusters are stable.

## 4 SIMILARITY-AWARE DIFFUSION MODEL-BASED IMPUTATION (SADI)

Consider a random variable $\mathbf{x}_0$, where the subscript $s$ represents a step in a Markov process. Here, $s = 0$ specifically refers to the variables $\mathbf{x}_0$ originating from the true data distribution $q(\mathbf{x}_0)$, which is our primary interest. We consider a setting where the data distribution is a mixture distribution with $K$ components that can be represented as

$$q(\mathbf{x}_0) = c_1 q_1(\mathbf{x}_0) + \cdots + c_K q_K(\mathbf{x}_0). \tag{6}$$

Each component corresponds to a group or class of the data, and the sum of the weights of each group $c_k$ equals 1.

In the traditional CSDI algorithm, a batch of random samples is drawn from the overall data distribution $q(\mathbf{x}_0)$ at each optimization step. Consequently, each sample in the batch may come from different groups, limiting the amount of shared information across samples due to their dissimilarity. In contrast, the similarity-aware diffusion model utilizes the decomposition from eq. (6) and draws a batch of random samples from the same group $q_k(\mathbf{x}_0)$ at each optimization step, ensuring that samples within the same batch are similar. This sample similarity introduces an additional source of information that the model can learn from, thereby enhancing imputation performance. We name the optimized imputation method as similarity-aware diffusion model-based imputation (SADI). For complex EHR time series data, defining groups and patient similarity can be challenging. To address this issue, we have developed a novel data-driven approach to evaluate group information and patient similarity, which is detailed in Section 5.1.

We choose to use a diffusion model to approximate the conditional distribution $q(\mathbf{x}_0^{ta}|\mathbf{x}_0^{co})$. Specifically, the diffusion model contains one forward process eq. (1), and one reverse process eq. (3). Intuitively, the forward process defines a sequence of latent variables $\mathbf{x}_1^{ta}, \dots, \mathbf{x}_S^{ta}$ through a Markov chain (see eq. (1)). As the step $s$ increases, the determined part $\sqrt{\bar{\alpha}_s}\mathbf{x}_0^{ta}$ in the latent variable $\mathbf{x}_s^{ta}$ decreases, while the noise part $\sqrt{1-\bar{\alpha}_s}\boldsymbol{\epsilon}$ increases, as illustrated in eq. (2). Eventually, the last latent variable $\mathbf{x}_S^{ta}$ is approximately to be random noise. On the contrary, the reverse process aims to denoise $\mathbf{x}_S^{ta}$ and recover the original data $\mathbf{x}_0^{ta}$. The reverse process eq. (3) is defined by a learnable Markov chain, where the transition probability follows a normal distribution and the corresponding mean and variance are learned by models.

The diffusion model is usually trained by optimizing the variational bound on the negative log-likelihood (Sohl-Dickstein et al., 2015). By considering DDPM parameterization eq. (4), training the diffusion model is essentially training the denoising function $\boldsymbol{\epsilon}_\theta$ (Ho et al., 2020; Tashiro et al., 2021), which is represented by a deep neural network with parameter $\theta$. The loss function of similarity-aware diffusion model is

$$\begin{aligned}&\min_{\theta} \mathcal{L}(\theta)\\ :=&\min_{\theta} \mathbb{E}_{k\sim\mathcal{C}\{c_1,c_2,\dots,c_K\},\mathbf{x}_0\sim q_k(\mathbf{x}_0),\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),s\sim\mathcal{U}\{1,S\}}\\ &\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta(\mathbf{x}_s^{ta},s|\mathbf{x}_0^{co})\|_2^2\end{aligned} \tag{7}$$

Here $\mathcal{C}\{c_1, c_2, \dots, c_K\}$ represents a categorical distribution, and the group number $k$ is sampled from $\{1, 2, \dots, K\}$ with corresponding probability $\{c_1, c_2, \dots, c_K\}$. As we can observe, the distinction between the SADI loss function eq. (7) and CSDI Equation (5) lies in the sampling approach. SADI first determines the group number and then samples data from the corresponding group distribution, whereas CSDI directly samples data from the entire distribution. Thus, when optimizing the model using a batch of samples, SADI draws a batch of similar samples at each optimization step, while in CSDI, samples may be dissimilar since they could be drawn from different groups.

As we mentioned in section 3.1, the target part should be known during the training phase to calculate the loss function. So the target part should be selected from observed values during training. Once the denoising function $\boldsymbol{\epsilon}_\theta$ is trained, we can sample imputations for $\mathbf{x}_0^{mis}$ from the reverse process eq. (3) by letting the target part be all the missing values and the conditional part be all the observed values. The training and sampling algorithms are presented in algorithm 1 and algorithm 2 respectively.

## 5 SADI for INCOMPLETE TEMPORAL EHR DATA

In this section, we describe the procedure of how to use SADI to impute incomplete temporal EHR data $\{\mathbf{X}_i\}_{i=1}^n$. The motivation for applying SADI is to effectively incorporate information across similar patients and model the correlations among them to further improve imputation performance.

### 5.1 The overall imputation pipeline of SADI

Due to the incompleteness of EHR data, it is challenging to access patient similarity directly. So we propose a procedure as shown in Figure 1, which entails four steps. The first step is to utilize an imputation method, such as MICE or CSDI, to perform an initial imputation and obtain the complete dataset $\{\bar{\mathbf{X}}_i\}_{i=1}^n$. The second step is to apply a time series clustering algorithm, such as the K-SC clustering method, to divide $\{\bar{\mathbf{X}}_i\}_{i=1}^n$ into $K$ groups $\{\bar{\mathbf{X}}_i\}_{i\in G_k}$ for $k = 1, \dots, K$. Here, the number of groups $K$ is a predefined hyperparameter, and the goal of the second step is to obtain the group information $G_k$. We then partition the original dataset $\{\mathbf{X}\}_{i=1}^n$ into the corresponding groups $\mathbf{X}_{G_k} = \{\mathbf{X}_i\}_{i\in G_k}$ for $k = 1, \dots, K$. Then we regard each group $\mathbf{X}_{G_k}$ containing samples from the distribution $q_k(x_0)$ defined in section 4 and the samples in the same group are similar. The third step is to leverage the SADI framework in section 5 for approximating the distribution of missing values conditioned on observed values. In this step, we propose a novel patient-similarity-aware denoising function $\boldsymbol{\epsilon}_\theta$ and train the denoising function $\boldsymbol{\epsilon}_\theta$ on groups $\{\mathbf{X}_{G_k}\}_{k=1}^K$. The fourth and last step is to utilize the trained denoising function $\boldsymbol{\epsilon}_\theta$ to sample imputation from the reverse process for missing data. Note that subscript $s$ in $\mathbf{X}_{i,s}$ represents the $s$-th step in the Markov process in the remainder of section 5, and $\mathbf{X}_{i,0} = \mathbf{X}_i$ denotes data from the true data distribution.

### 5.2 Similarity-aware denoising function

We first specify the input and the structure of the patient-similarity-aware denoising function $\boldsymbol{\epsilon}_\theta$. Recall that the original denoising function $\boldsymbol{\epsilon}_\theta(\mathbf{x}_s^{ta}, s|\mathbf{x}_0^{co})$ in section 3.1 takes noisy target part $\mathbf{x}_s^{ta}$, step $s$ and conditional part $\mathbf{x}_0^{co}$ as input and predicts the noise contained in the noisy target $\mathbf{x}_s^{ta}$. During the training phase, a random batch of training data from $q(\mathbf{x}_0)$ is used to optimize the original denoising function. SADI refines this process to use information across similar patients. Specifically, the denoising function $\boldsymbol{\epsilon}_\theta(\mathbf{x}_s^{ta}, s|\mathbf{x}_0^{co})$ takes a batch of samples from $q_k(\mathbf{x}_0)$ as input. Since we have regarded each group data $\mathbf{X}_{G_k}$ as samples from $q_k(\mathbf{x}_0)$ after the clustering step. So the denoising function $\boldsymbol{\epsilon}_\theta(\mathbf{x}_s^{ta}, s|\mathbf{x}_0^{co})$ of SADI takes a batch of similar patients' data $\mathbf{X}_B$ as input, where $B \subset G_k$ with batch size $|B| = b$. To handle the varying shapes of the target part $\mathbf{X}_{i,s}^{ta}$ and conditional part $\mathbf{X}_{i,0}^{co}$ of patients' EHR data, zero padding is applied to both parts to ensure that they have the same shape $p \times T$. As such, a conditional mask $\mathbf{M}_i^{co} \in \{0, 1\}^{p\times T}$ is also passed as input to indicate the position of conditional observations. We also apply zero padding to the outputs to keep the output also lying in the sample space $\mathbb{R}^{p\times T}$. Denote the stacked target parts, the stacked conditional parts, and the stacked masks of batched data from the $k$-th group by

$$\begin{aligned}\mathbf{X}_{B,s}^{ta} &= \mathrm{Stack}(\{\mathbf{X}_{i,s}^{ta}\}_{i\in B}) \in \mathbb{R}^{b\times p\times T}\\ \mathbf{X}_{B,0}^{co} &= \mathrm{Stack}(\{\mathbf{X}_{i,0}^{co}\}_{i\in B}) \in \mathbb{R}^{b\times p\times T}\\ \mathbf{M}_{B}^{co} &= \mathrm{Stack}(\{\mathbf{M}_{i}^{co}\}_{i\in B}) \in \mathbb{R}^{b\times p\times T}\end{aligned}$$

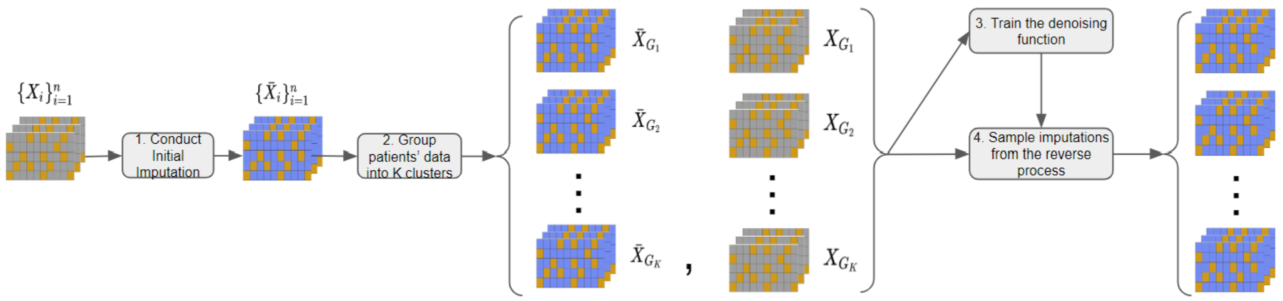
$\{X_i\}_{i=1}^n$
1. Conduct Initial Imputation
$\{\bar{X}_i\}_{i=1}^n$
2. Group patients' data into K clusters
$\bar{X}_{G_1}$
$\bar{X}_{G_2}$
$\bar{X}_{G_K}$
$X_{G_1}$
$X_{G_2}$
$X_{G_K}$
3. Train the denoising function
4. Sample imputations from the reverse process

Figure 1: The procedure of temporal missing value imputation with SADI. Each table represents a patient's EHR data, with gray cells indicating missing values, orange cells indicating observed values, and blue cells indicating imputed values. The process involves (1) conducting an initial imputation (e.g., using CSDI) on the entire dataset $\{\mathbf{X}_i\}_{i=1}^n$ to generate imputed data $\{\bar{\mathbf{X}}_i\}_{i=1}^n$, (2) applying a clustering algorithm (e.g., K-SC) to categorize imputed data $\{\bar{\mathbf{X}}_i\}_{i=1}^n$ into K clusters $\{\bar{\mathbf{X}}_{G_k}\}_{k=1}^K$ and acquiring the corresponding original data clusters $\{\mathbf{X}_{G_k}\}_{k=1}^K$, (3) training SADI on the K clustered datasets $\{\mathbf{X}_{G_k}\}_{k=1}^K$, and (4) sampling imputations for the data clusters using the reverse process of SADI.

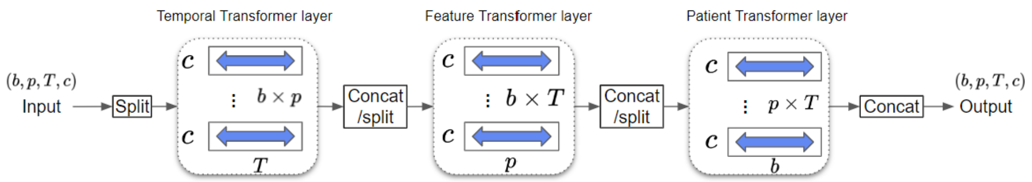

Figure 2: Architecture of 3D attention mechanism. Given a tensor from the $k$-th group with $b$ patients, $p$ features, $T$ time points, and $c$ channels, the temporal transformer layer processes inputs of shape $(1, 1, T, c)$ to learn temporal dependency. The feature transformer layer processes inputs of shape $(1, p, 1, c)$ to learn feature dependency. The patient transformer layer processes inputs of shape $(b, 1, 1, c)$ to learn patient dependency.

Then the dimensions of the input and output of the denoising function $\boldsymbol{\epsilon}_\theta(\mathbf{X}_{B,s}^{ta}, s|\mathbf{X}_{B,0}^{co}, \mathbf{M}_B)$ can be written as: $(\mathbb{R}^{b\times p\times T} \times \mathbb{R}|\mathbb{R}^{b\times p\times T} \times \mathbb{R}^{b\times p\times T}) \to \mathbb{R}^{b\times p\times T}$. Note that the same diffusion step $s$ is applied to all the patients' data in the same batch $B \subset G_k$. This is different from the training procedure of the original denoising function in section 3.1, where each training sample has its own diffusion step $s$.

The structure of our patient-similarity-aware denoising function $\boldsymbol{\epsilon}_\theta$ is designed using techniques from DiffWave (Kong et al., 2020) and CSDI (Tashiro et al., 2021) which consist of multiple residual layers with $c$ residual channels. The details of the denoising function can be found in appendix A.3. Here we discuss the main difference from previous works, which is the use of a three-dimensional attention mechanism within each residual layer to learn the temporal, feature, and patient dependencies, as shown in Figure 2. This is achieved by incorporating three transformer layers, each with a single-layer transformer encoder. The first transformer layer captures temporal dependencies by processing input tensors for each feature and patient, where the length of each input sequence is $T$. The second transformer layer learns feature dependencies by operating on input tensors for each time point and patient. Lastly, the third transformer layer captures patient dependencies by processing input tensors for each time point and feature.

## 6 EXPERIMENTS

### 6.1 Datasets

To evaluate the performance of SADI in comparison with the current SOTA, we conduct numerical experiments on two real-world EHR datasets: a dataset from The Critical Path For Alzheimer's Disease (CPAD) consortium[1] (Sivakumaran et al., 2020) and the PhysioNet Challenge 2012[2] dataset (Silva et al., 2012). For both datasets, we run each experiment five times. Our main focus is on the CPAD dataset, a temporal EHR dataset collected from a non-ICU setting. The PhysioNet dataset, on the other hand, is a high-frequency time series dataset collected from ICUs. Both datasets have been anonymized and do not contain sensitive in-

[1]See `https://c-path.org/programs/cpad/`
[2]See `https://PhysioNet.org/content/challenge-2012/1.0.0/`

| | CPAD | PhysioNet Challenge 2012 |
|---|---|---|
| Number of patients | 11369 | 4000 |
| Number of features | 37 | 35 |
| Number of time points | 12 | 48 |
| Percentage of missing values | 79% | 80% |

Table 1: Description of two EHR datasets

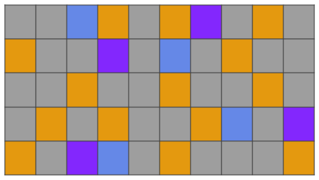

Figure 3: An illustration of dividing a patient's EHR data with 5 features and 10 time points into training, validation, and testing sets. Each row represents a feature and each column represents a time point. Missing values are shown in gray, while observed values are divided into orange (training), blue (validation), and purple (testing) segments.

formation.

The CPAD dataset consists of 36 sub-tables, including clinical events, lab results, imaging results, and other data domains collected over a period of up to four years. In our experiments, we focus on the lab result table and choose the most frequent 37 features. This table includes 11369 patients. We preprocess the dataset to monthly-based EHR data with 12 time points. Only the visit data within the first year after the initial visit are used to generate the EHR data. This evenly spaced EHR dataset has around 79% missing values in total.

The PhysioNet dataset contains 4000 patients' clinical multivariate time series data from ICUs. Except for the general descriptors like age and gender, each multivariate time series has 35 features, including Glucose, DiasABP, and so on. Those features are irregularly sampled in the first 48 hours after admission to the ICU. We preprocess the original dataset to hourly-based time series with 48 time points. This evenly spaced time series dataset has around 80% missing values in total. The description of these two datasets are summarized in Table 1, and more details can be found in appendix A.1.

Since there are no ground truths for missing values on the CPAD data and the PhysioNet data, we artificially mask out 10% of observed values as test data to evaluate model performance under three missing data mechanisms, MCAR, MAR, and MNAR (Little and Rubin, 2019). The remaining observed data are used as the training data, and we randomly select 10% of the training data as the validation data, as shown in Figure 3.

## 6.2 Missing mechanism

Briefly, MCAR occurs when the missingness is independent of both the observed and missing values. MAR occurs when the missingness depends solely on the observed values. Lastly, MNAR occurs when the missingness is dependent on both observed and missing values. MCAR typically is not valid in EHR data, whereas MAR is more plausible. Imputing missing values in MNAR settings is more challenging than in MCAR and MAR settings. It is well known that one cannot test MAR vs MNAR using observed data. As such, to gain a comprehensive assessment of SADI in comparison with the current SOTA, we evaluate their performance under all three missing data mechanisms. The detailed description can be found in appendix A.2.

## 6.3 Methods to be compared

We evaluate our proposed SADI with both RNN-based and attention-based imputation models which are listed below. All models are trained with GPU RTX 2080. Experiment details can be found in appendix A.3. To conduct accurate imputation, SADI takes the median of 100 generated samples as the final imputation. **RNN-based methods:** (1) **MRNN** (Yoon et al., 2018) uses bidirectional LSTM to impute missing values and models both temporal and feature dependencies. (2) **RITS** (Cao et al., 2018) is a simplified version of BRITS and only models the temporal dependency. (3) **BRITS** (Cao et al., 2018) is similar to MRNN, and also uses bidirectional RNN to model both temporal and feature dependencies. **Transformer-based methods:** (4) **CSDI** (Tashiro et al., 2021) is based on diffusion models and utilizes two transformer layers to capture time and feature dependencies.

## 6.4 Performance metrics

We use three metrics to evaluate the imputation performance: mean absolute error (MAE), mean relative error (MRE) and root mean square error (RMSE). Suppose $\texttt{target}_i$ is the ground truth for the $i$-th item and $\texttt{estimation}_i$ is the predictive value for the $i$-th item, and there are $N$ items in total. Then three metrics are defined as follows

$$\texttt{MAE} = \frac{\sum_i |\texttt{estimation}_i - \texttt{target}_i|}{N}$$

$$\texttt{MRE} = \frac{\sum_i |\texttt{estimation}_i - \texttt{target}_i|}{\sum_i |\texttt{target}_i|}$$

$$\texttt{RMSE} = \sqrt{\frac{\sum_i |\texttt{estimation}_i - \texttt{target}_i|^2}{N}}$$

| Methods | MCAR | | | MAR | | | MNAR | | |
|---|---|---|---|---|---|---|---|---|---|
| | MAE | RMSE | MRE | MAE | RMSE | MRE | MAE | RMSE | MRE |
| MRNN | 0.233(0.001) | 0.478(0.013) | 33.5%(0.001) | 0.233(0.001) | 0.672(0.030) | 38.5%(0.001) | 0.246(0.001) | 0.723(0.123) | 44.4%(0.003) |
| RITS | 0.259(0.001) | 0.547(0.009) | 40.3%(0.002) | 0.252(0.001) | 0.654(0.024) | 41.8%(0.001) | 0.265(0.002) | 0.730(0.122) | 47.7%(0.002) |
| BRITS | 0.239(0.001) | 0.532(0.008) | 37.1%(0.001) | 0.233(0.001) | 0.629(0.019) | 38.7%(0.001) | 0.244(0.001) | 0.710(0.117) | 44.1%(0.003) |
| CSDI | 0.206(0.006) | 0.440(0.004) | 30.2%(0.009) | 0.207(0.002) | 0.640(0.025) | 34.3%(0.003) | 0.219(0.001) | 0.679(0.134) | 39.5%(0.003) |
| SADI | **0.190**(0.001) | **0.416**(0.005) | **28.0%**(0.002) | **0.194**(0.002) | **0.624**(0.031) | **32.1%**(0.003) | **0.206**(0.002) | **0.664**(0.137) | **37.1%**(0.005) |

Table 2: Performance comparison of methods on the CPAD dataset under MCAR, MAR, and MNAR mechanisms. We report the mean and standard error for five trials.

| Methods | MCAR | | | MAR | | | MNAR | | |
|---|---|---|---|---|---|---|---|---|---|
| | MAE | RMSE | MRE | MAE | RMSE | MRE | MAE | RMSE | MRE |
| MRNN | 0.381(0.001) | 0.721(0.096) | 54.6%(0.001) | 0.377(0.001) | 0.677(0.013) | 48.3%(0.001) | 0.389(0.001) | 0.782(0.029) | 47.8%(0.001) |
| RITS | 0.322(0.001) | 0.653(0.074) | 44.9%(0.001) | 0.323(0.001) | 0.654(0.012) | 39.9%(0.001) | 0.329(0.001) | 0.795(0.027) | 40.3%(0.001) |
| BRITS | 0.294(0.001) | 0.632(0.056) | 41.7%(0.001) | 0.296(0.001) | 0.621(0.008) | 37.9%(0.001) | 0.313(0.001) | 0.764(0.019) | 38.2%(0.001) |
| CSDI | 0.242(0.001) | **0.586**(0.043) | **34.3%**(0.002) | 0.243(0.001) | 0.546(0.007) | 31.0%(0.001) | 0.252(0.001) | 0.689(0.019) | 30.7%(0.001) |
| SADI | **0.241**(0.001) | 0.624(0.069) | **34.3%**(0.002) | **0.241**(0.001) | **0.543**(0.007) | **30.8%**(0.001) | **0.248**(0.001) | **0.681**(0.023) | **30.1%**(0.001) |

Table 3: Performance comparison of methods on the PhysioNet dataset under MCAR, MAR, and MNAR mechanisms. We report the mean and standard error for five trials.

### 6.5 Experiment results

We first evaluate the performance of SADI and four other SOTA methods on the CPAD dataset (the non-ICU dataset) and compare their imputation performance. The results under MCAR, MAR, and MNAR settings are presented in Table 2. The best performance in each table is highlighted in bold. First, the three settings show similar results in terms of comparisons across the methods. Particularly, three RNN-based methods (MRNN, RITS, and BRITS) tend to perform worse than the two transformer-based methods (CSDI and SADI), suggesting that transformer layers are better suited for modeling sequence dependencies. Among the transformer-based methods, SADI significantly outperforms CSDI in all scenarios. Specifically, SADI reduces MAE, RMSE, and MRE by around $8\%, 6\%$, and $7\%$, respectively, compared to CSDI in the MCAR setting. Similar improvements are also observed in the MAR setting and the MNAR setting, where SADI reduces the MAE, RMSE, and MRE by 6.5%, 2.5%, 8% and 6%, 2%, 6% respectively, compared to CSDI. This suggests that it is insufficient to only model temporal dependency and feature dependency on sparse temporal EHR data. Additionally, the consistent better performance of SADI over CSDI under all missing data mechanisms also highlights the importance of modeling dependency among similar patients, which makes the model more robust and able to handle different types of missing data mechanisms in EHR data. Ablation study on the CPAD dataset can be found in appendix A.4.

Then we evaluate SADI in comparison with the other methods on the PhysioNet dataset (the ICU dataset) under MCAR, MAR, and MNAR mechanisms, as shown in Table 3. The results still show that the two transformer-based methods (CSDI and SADI) outperform the three RNN-based methods (MRNN, RITS, and BRITS). But the improvement of SADI over CSDI is more modest, at less than 2% in most cases. As discussed earlier, in the ICU setting, the strong temporal correlation between time points could provide sufficient information to achieve accurate imputation. In this case, borrowing information across similar patients only offers marginal improvement. Nevertheless, our proposed SADI still achieves the best or close to the best performance in all settings.

## 7 CONCLUSION/DISCUSSION

In this work, we present a new imputation method, SADI, for imputing missing values in temporal EHR data. SADI enables borrowing information across similar patients, in addition to leverage information across time and across features, to fill in missing values. Our experiments show that SADI outperforms current SOTA EHR data imputation methods in temporal EHR data from non-ICU settings and still achieves the best or close to the best performance in EHR data from ICUs. One limitation of SADI is that the clustering step can be computationally expensive when applied to large datasets with a large number of patients. To mitigate this issue, a potential future research direction would be to generate embedding for EHR data and cluster the dataset based on embeddings (Nalmpantis and Vrakas, 2019; Shukla and Marlin, 2021).

**Acknowledgements**

This work was supported in part by National Institutes of Health grants, R01GM124111, RF1AG063481, and U01CA274576. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

## References

Berndt, D. J. and Clifford, J. (1994). Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, USA:.

Cao, W., Wang, D., Li, J., Zhou, H., Li, L., and Li, Y. (2018). Brits: Bidirectional recurrent imputation for time series. *Advances in neural information processing systems*, 31.

Che, Z., Purushotham, S., Cho, K., Sontag, D., and Liu, Y. (2018). Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):1–12.

Chen, N., Zhang, Y., Zen, H., Weiss, R. J., Norouzi, M., and Chan, W. (2020). Wavegrad: Estimating gradients for waveform generation. *arXiv preprint arXiv:2009.00713*.

Cowie, M. R., Blomster, J. I., Curtis, L. H., Duclaux, S., Ford, I., Fritz, F., Goldman, S., Janmohamed, S., Kreuzer, J., Leenay, M., et al. (2017). Electronic health records to facilitate clinical research. *Clinical Research in Cardiology*, 106:1–9.

Fortuin, V., Baranchuk, D., Rätsch, G., and Mandt, S. (2020). Gp-vae: Deep probabilistic time series imputation. In *International conference on artificial intelligence and statistics*, pages 1651–1661. PMLR.

Getzen, E., Ungar, L., Mowery, D., Jiang, X., and Long, Q. (2023). Mining for equitable health: Assessing the impact of missing data in electronic health records. *Journal of Biomedical Informatics*, page 104269.

Gianfrancesco, M. A., Tamang, S., Yazdany, J., and Schmajuk, G. (2018). Potential biases in machine learning algorithms using electronic health record data. *JAMA internal medicine*, 178(11):1544–1547.

Häyrinen, K., Saranto, K., and Nykänen, P. (2008). Definition, structure, content, use and impacts of electronic health records: a review of the research literature. *International journal of medical informatics*, 77(5):291–304.

Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851.

Jensen, P. B., Jensen, L. J., and Brunak, S. (2012). Mining electronic health records: towards better research applications and clinical care. *Nature Reviews Genetics*, 13(6):395–405.

Kadkhodaie, Z. and Simoncelli, E. P. (2020). Solving linear inverse problems using the prior implicit in a denoiser. *arXiv preprint arXiv:2007.13640*.

Khojandi, A., Tansakul, V., Li, X., Koszalinski, R. S., and Paiva, W. (2018). Prediction of sepsis and in-hospital mortality using electronic health records. *Methods of information in medicine*, 57(04):185–193.

Kong, Z., Ping, W., Huang, J., Zhao, K., and Catanzaro, B. (2020). Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*.

Lauritsen, S. M., Kalør, M. E., Kongsgaard, E. L., Lauritsen, K. M., Jørgensen, M. J., Lange, J., and Thiesson, B. (2020). Early detection of sepsis utilizing deep learning on electronic health record event sequences. *Artificial Intelligence in Medicine*, 104:101820.

Little, R. J. and Rubin, D. B. (2019). *Statistical analysis with missing data*, volume 793. John Wiley & Sons.

Luo, Y., Cai, X., Zhang, Y., Xu, J., et al. (2018). Multivariate time series imputation with generative adversarial networks. *Advances in neural information processing systems*, 31.

Luo, Y., Zhang, Y., Cai, X., and Yuan, X. (2019). E2gan: End-to-end generative adversarial network for multivariate time series imputation. In *Proceedings of the 28th international joint conference on artificial intelligence*, pages 3094–3100. AAAI Press.

Mittal, G., Engel, J., Hawthorne, C., and Simon, I. (2021). Symbolic music generation with diffusion models. *arXiv preprint arXiv:2103.16091*.

Nalmpantis, C. and Vrakas, D. (2019). Signal2vec: Time series embedding representation. In *International conference on engineering applications of neural networks*, pages 80–90. Springer.

Niu, C., Song, Y., Song, J., Zhao, S., Grover, A., and Ermon, S. (2020). Permutation invariant graph generation via score-based generative modeling. In *International Conference on Artificial Intelligence and Statistics*, pages 4474–4484. PMLR.

Ozer, M., Sapienza, A., Abeliuk, A., Muric, G., and Ferrara, E. (2020). Discovering patterns of online popularity from time series. *Expert Systems with Applications*, 151:113337.

Paparrizos, J. and Gravano, L. (2015). k-shape: Efficient and accurate clustering of time series. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pages 1855–1870.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch: An imperative style, high-performance deep learning library.

*Advances in neural information processing systems*, 32.

Rajkomar, A., Hardt, M., Howell, M. D., Corrado, G., and Chin, M. H. (2018a). Ensuring fairness in machine learning to advance health equity. *Annals of internal medicine*, 169(12):866–872.

Rajkomar, A., Oren, E., Chen, K., Dai, A. M., Hajaj, N., Hardt, M., Liu, P. J., Liu, X., Marcus, J., Sun, M., et al. (2018b). Scalable and accurate deep learning with electronic health records. *NPJ digital medicine*, 1(1):18.

Salvador, S. and Chan, P. (2007). Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580.

Shukla, S. N. and Marlin, B. M. (2021). Multi-time attention networks for irregularly sampled time series. *arXiv preprint arXiv:2101.10318*.

Silva, I., Moody, G., Scott, D. J., Celi, L. A., and Mark, R. G. (2012). Predicting in-hospital mortality of icu patients: The physionet/computing in cardiology challenge 2012. In *2012 Computing in Cardiology*, pages 245–248. IEEE.

Sivakumaran, S., Romero, K., Hanan, N. J., Sinha, V., Haeberlein, S. B., and Gold, M. (2020). The critical path for alzheimer's disease (cpad): Precompetitive data sharing and generation of innovative high-impact quantitative tools to support alzheimer's disease drug development: Human/trial design. *Alzheimer's & Dementia*, 16:e043919.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR.

Song, Y. and Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. (2020). Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.

Steele, A. J., Denaxas, S. C., Shah, A. D., Hemingway, H., and Luscombe, N. M. (2018). Machine learning models in electronic health records can outperform conventional survival models for predicting patient mortality in coronary artery disease. *PloS one*, 13(8):e0202344.

Suo, Q., Zhong, W., Xun, G., Sun, J., Chen, C., and Zhang, A. (2020). Glima: Global and local time series imputation with multi-directional attention learning. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 798–807. IEEE.

Tashiro, Y., Song, J., Song, Y., and Ermon, S. (2021). Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34:24804–24816.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Wells, B. J., Chagin, K. M., Nowacki, A. S., and Kattan, M. W. (2013). Strategies for handling missing data in electronic health record derived data. *Egems*, 1(3).

Yang, J. and Leskovec, J. (2011). Patterns of temporal variation in online media. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 177–186.

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

## Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Not Applicable]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

   (a) Citations of the creator If your work uses existing assets. [Yes]

   (b) The license information of the assets, if applicable. [Not Applicable]

   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes]

   (d) Information about consent from data providers/curators. [Yes]

   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Yes]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

   (a) The full text of instructions given to participants and screenshots. [Not Applicable]

   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

## A Experiment details

### A.1 Dataset details

**Physinet 2012 dataset** 35 features are selected: Albumin, ALP, ALT, AST, Bilirubin, BUN, Cholesterol, Creatinine, DiasABP, FiO2, GCS, Glucose, HCO3, HCT, HR, K, Lactate, Mg, MAP, Na, NIDiasABP, NIMAP, NISysABP, PaCO2, PaO2, pH, Platelets, RespRate, SaO2, SysABP, Temp, TroponinI, TroponinT, Urine, WBC

**Critical Path For Alzheimer's Disease (CPAD) dataset** 37 features are selected: Hemoglobin, Alkaline Phosphatase, Creatinine, Alanine Aminotransferase, Aspartate Aminotransferase, Potassium, Sodium, Gamma Glutamyl Transferase, Albumin, Cholesterol, Calcium, Leukocytes, Triglycerides, Blood Urea Nitrogen, Glucose, Bilirubin, Platelets, Eosinophils/Leukocytes, Lymphocytes/Leukocytes, Monocytes/Leukocytes, Basophils/Leukocytes, Neutrophils/Leukocytes, Hematocrit, Creatine Kinase, Bicarbonate, Prothrombin Intl. Normalized Ratio, Activated Partial Thromboplastin Time, C Reactive Protein, Chloride, Protein, Erythrocytes, Monocytes, Basophils, Lymphocytes, Neutrophils, Eosinophils, Indirect Bilirubin

### A.2 Missing mechanism

For the MCAR setting, we set each observed entry to have the same probability of 0.1 to be masked out as a part of the test set. For the MAR and the MNAR settings, the specific details of how missingness is introduced are presented below.

Recall that the observations are denoted by $\mathbf{X}^d_{i,t}$ ($i = 1, \ldots, n$; $t = 1, \ldots, T$; $d = 1, \ldots, p$). We also denote the original missing indicator $\mathbf{M}^d_{i,t}$ as ${}_1\mathbf{M}^d_{i,t}$. Denote the missing indicator after masking out the test set as ${}_2\mathbf{M}^d_{i,t}$. Here we describe how to generate ${}_2\mathbf{M}^d_{i,t}$ based on ${}_1\mathbf{M}^d_{i,t}$. Specifically, if MAR mechanism is applied, ${}_2\mathbf{M}^d_{i,t}$ is generated through the following rules:

$$
\begin{aligned}
&P({}_2\mathbf{M}^d_{i,t} = 0) \\
&= \begin{cases} 1, & \text{if } {}_1\mathbf{M}^d_{i,t} = 0 \\ \frac{p^t \cdot (\sum_{d'=1}^{D} {}_1\mathbf{M}^{d'}_{i,t}) \cdot e^{-\sum_{t'<t} \omega_{t'} \cdot \mathbf{X}^d_{i,t'} \cdot {}_2\mathbf{M}^d_{i,t'}}}{\sum_{d'=1}^{D} {}_1\mathbf{M}^{d'}_{i,t} \cdot e^{-\sum_{t'<t} \omega_{t'} \cdot \mathbf{X}^{d'}_{i,t'} \cdot {}_2\mathbf{M}^{d'}_{i,t'}}}, & \text{otherwise} \end{cases}
\end{aligned}
$$

Alternatively, if MNAR mechanism is applied, ${}_2\mathbf{M}^d_{i,t}$

is generated from ${}_1\mathbf{M}^d_{i,t}$ through the following rules:

$$
\begin{aligned}
&P({}_2\mathbf{M}^d_{i,t} = 0) \\
&= \begin{cases} 1, & \text{if } {}_1\mathbf{M}^d_{i,t} = 0 \\ \frac{p^t \cdot (\sum_{d'=1}^{D} {}_1\mathbf{M}^{d'}_{i,t}) \cdot e^{-\sum_{t' \le t} \omega_{t'} \cdot \mathbf{X}^d_{i,t'} \cdot {}_1\mathbf{M}^d_{i,t'}}}{\sum_{d'=1}^{D} {}_1\mathbf{M}^{d'}_{i,t} \cdot e^{-\sum_{t' \le t} \omega_{t'} \cdot \mathbf{X}^{d'}_{i,t'} \cdot {}_1\mathbf{M}^{d'}_{i,t'}}}, & \text{otherwise} \end{cases}
\end{aligned}
$$

Here $p^t$ denotes the proportion of observed data that are masked out as the test dataset at time step $t$, and $\omega_{t'}$ are sampled from $U(0, 1)$ (but only sampled once for the entire dataset). In our experiments, $p^t$ is set to 10% for all $t$.

### A.3 Implementation details

All experiments are repeated five times under the random seeds from 42 to 46. For RITS[3], BRITS[4], and CSDI[5], we use the open-access implementations provided by their authors. For MRNN[6], we use open-access implementations provided by the author of BRITS. We use the default or the recommended hyperparameters of RITS, BRITS, and MRNN in their papers on both the PhysioNet challenge 2012 data and CPAD data. For CSDI, we use the default parameter on the PhysioNet challenge 2012 data and fine-tune the batch size of CSDI based on validation MAE on the CPAD dataset for a fair comparison.

For SADI, we use CSDI with the default parameters to generate the initial imputation in the first step. Then we use the K-SC method with `shift=5` to conduct the clustering step. We show the structure of the denoising function in Figure 4. The number of residual layers is 4, the batch size is 32 and the residual channel is 64. Each transformer layer used in Figure 4 is a 1-layer TransformerEncoder implemented in PyTorch (Paszke et al., 2019), and it consists of a multi-head attention layer, fully-connected layers, and layer normalization. The number of heads in each attention layer is 8. By following previous works (Vaswani et al., 2017; Kong et al., 2020; Tashiro et al., 2021), we use the following 128-dimensions embedding for the diffusion step $s$:

$$
\begin{aligned}
s_{embedding}(s) = (&\sin(10^{0 \cdot 4/63} s), \ldots, \sin(10^{63 \cdot 63/63} s), \\
&\cos(10^{0 \cdot 4/63} s), \ldots, \cos(10^{63 \cdot 63/63} s))
\end{aligned}
$$

Similarly, we adopt the following 128-dimensions embedding for the time point $t$ as side information:

$$
\begin{aligned}
t_{embedding}(t) = (&\sin(t/\tau^{0/64}), \ldots, \sin(t/\tau^{63/64}), \\
&\cos(t/\tau^{0/64}), \ldots, \cos(t/\tau^{63/64}))
\end{aligned}
$$

[3]See `https://github.com/caow13/BRITS`
[4]See `https://github.com/caow13/BRITS`
[5]See `https://github.com/ermongroup/CSDI`
[6]See `https://github.com/caow13/BRITS`

| Temporal | Feature | Patient | MAE | RMSE | MRE |
|---|---|---|---|---|---|
| ✓ | ✓ | ✓ | 0.206(0.002) | 0.664(0.137) | 37.1%(0.005) |
| ✗ | ✓ | ✓ | 0.294(0.001) | 0.787(0.114) | 53.0%(0.003) |
| ✓ | ✗ | ✓ | 0.228(0.001) | 0.700(0.126) | 41.1%(0.002) |
| ✓ | ✓ | ✗ | 0.220(0.004) | 0.681(0.133) | 39.7%(0.008) |
| ✓ | ✗ | ✗ | 0.244(0.003) | 0.731(0.125) | 44.0%(0.006) |
| ✗ | ✓ | ✗ | 0.352(0.020) | 0.885(0.094) | 63.4%(0.035) |
| ✗ | ✗ | ✓ | 0.339(0.001) | 0.843(0.104) | 61.1%(0.002) |

Table 4: Ablation study on the CPAD dataset under **MNAR**: In this table, a check mark indicates the inclusion of the corresponding transformer layer, while a cross mark denotes its exclusion. We report the mean and standard error for five trials.

where $\tau = 10000$.

We set the number of training epochs as 200 and chose the Adam optimizer to update the parameters. The learning rate is 0.001 and decayed to 0.0001 and 0.00001 at 75% and 90% of the total epochs, respectively. We set the number of diffusion steps as $S = 50$ and the noise level is increased from $\beta_1 = 0.0001$ to $\beta_S = 0.5$. We adopt the quadratic schedule (Tashiro et al., 2021) for other noise levels:

$$
\beta_s = \left(\frac{S - s}{S - 1}\sqrt{\beta_1} + \frac{s - 1}{S - 1}\sqrt{\beta_S}\right)
$$

### A.4 Ablation study on CPAD dataset

In our implementation of the denoising function $\boldsymbol{\epsilon}_\theta$, we design a 3D attention mechanism to learn the temporal, feature, and patient dependencies. In this section, we explore the contribution of each transformer layer using ablation. We show the result in Table 4. It shows that all three transformer layers contribute to the best final performance. Even in non-ICU settings, the temporal correlation is still the most important source of information that the model can learn from. Compared the first and fourth lines of the table, adding a patient layer improves the MAE, RMSE, and MRE performance by 6%, 2.5%, and 7%, respectively.

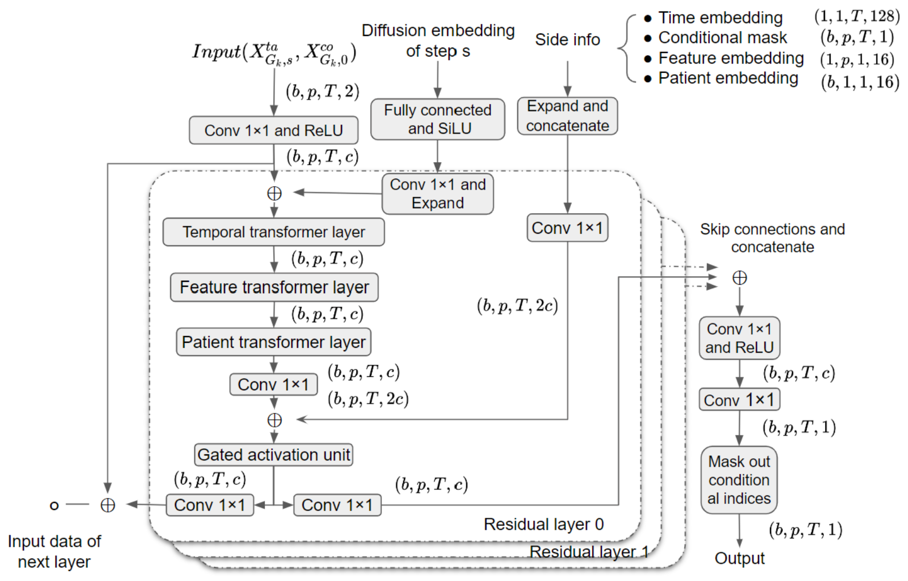
$Input(X^{ta}_{G_k,s}, X^{co}_{G_k,0})$
Diffusion embedding of step s
Side info
Time embedding $(1,1,T,128)$
Conditional mask $(b,p,T,1)$
Feature embedding $(1,p,1,16)$
Patient embedding $(b,1,1,16)$
$(b,p,T,2)$
Conv 1×1 and ReLU
Fully connected and SiLU
Expand and concatenate
$(b,p,T,c)$
Conv 1×1 and Expand
Temporal transformer layer
Conv 1×1
Skip connections and concatenate
Feature transformer layer
Patient transformer layer
$(b,p,T,2c)$
Conv 1×1 and ReLU
Gated activation unit
Conv 1×1
Mask out condition al indices
Residual layer 0
Residual layer 1
Input data of next layer
$(b,p,T,1)$
Output

Figure 4: structure of $\epsilon_\theta$