CSDI: Conditional Score-based Diffusion Models for Probabilistic Time Series Imputation

Anonymous Author(s)
Affiliation
Address
email

Abstract

The imputation of missing values in time series has many applications in healthcare and finance. While autoregressive models are natural candidates for time series imputation, score-based diffusion models have recently outperformed existing counterparts including autoregressive models in many tasks such as image generation and audio synthesis, and would be promising for time series imputation. In this paper, we propose Conditional Score-based Diffusion models for Imputation (CSDI), a novel time series imputation method that utilizes score-based diffusion models conditioned on observed data. Unlike existing score-based approaches, the conditional diffusion model is explicitly trained for imputation and can exploit correlations between observed values. On healthcare and environmental data, CSDI improves by 40-70% over existing probabilistic imputation methods on popular performance metrics. In addition, deterministic imputation by CSDI reduces the error by 5-20% compared to the state-of-the-art deterministic imputation methods. Furthermore, CSDI can also be applied to time series interpolation and probabilistic forecasting, and is competitive with existing baselines.

1 Introduction

Multivariate time series are abundant in real world applications such as finance, meteorology and healthcare. These time series data often contain missing values due to various reasons, including device failures and human errors [29, 38, 35]. Since missing values can hamper the interpretation of a time series, many studies have addressed the task of imputing missing values using machine learning techniques [19, 10, 2]. In the past few years, imputation methods based on deep neural networks have shown great success for both deterministic imputation [3, 4, 14] and probabilistic imputation [8]. These imputation methods typically utilize autoregressive models to deal with time series.

Score-based diffusion models – a class of deep generative models and generate samples by gradually converting noise into a plausible data sample through denoising – have recently achieved state-of-the-art sample quality in many tasks such as image generation [9, 33] and audio synthesis [12, 5], outperforming counterparts including autoregressive models. Diffusion models can also be used to impute missing values by approximating the scores of the posterior distribution obtained from the prior by conditioning on the observed values [33, 11]. While these approximations may work well in practice, they do not correspond to the exact conditional distribution.

In this paper, we propose CSDI, a novel probabilistic imputation method that directly learns the conditional distribution with *conditional* score-based diffusion models. Unlike existing score-based approaches, the conditional diffusion model is designed for imputation and can exploit useful information in observed values. We illustrate the procedure of time series imputation with CSDI in Figure 1. We start imputation from random noise on the left of the figure and gradually convert the noise into plausible time series through the reverse process p_θ of the conditional diffusion model. At

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

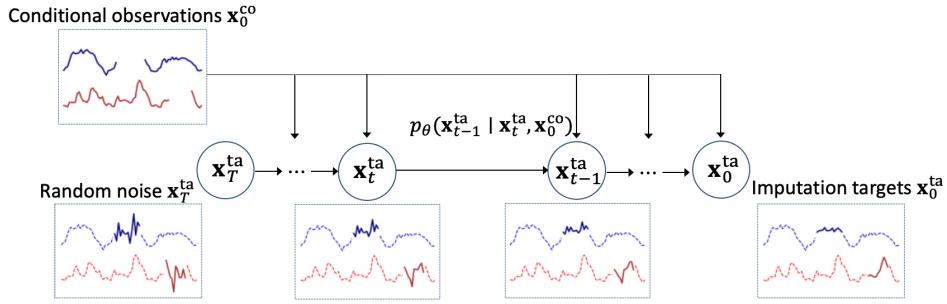

Figure 1: The procedure of time series imputation with CSDI. The reverse process p_θ gradually converts random noise into plausible time series, conditioned on observed values $\mathbf{x}_0^{\mathrm{co}}$. Dashed lines in each box represent observed values, which are plotted in order to show the relationship with generated imputation and not included in each $\mathbf{x}_t^{\mathrm{ta}}$.

each step t, the reverse process removes noise from the output of the previous step $(t+1)$. Unlike existing score-based diffusion models, the reverse process can take observations (on the top left of the figure) as a conditional input, allowing the model to exploit information in the observations for denoising. We utilize an attention mechanism to capture the temporal and feature dependencies of time series.

For training the conditional diffusion model, we need observed values (i.e., conditional information) and ground-truth missing values (i.e., imputation targets). However, in practice we do not know the ground-truth missing values, or training data may not contain missing values at all. Then, inspired by masked language modeling, we develop a self-supervised training method that separates observed values into conditional information and imputation targets. We note that CSDI is formulated for general imputation tasks, and is not restricted to time series imputation.

Our main contributions are as follows:

- We propose conditional score-based diffusion models for probabilistic imputation (CSDI), and implement CSDI for time series imputation. To train the conditional diffusion model, we develop a self-supervised training method.
- We empirically show that CSDI improves the continuous ranked probability score (CRPS) by 40-70% over existing probabilistic methods on healthcare and environmental data. Moreover, deterministic imputation with CSDI decreases the mean absolute error (MAE) by 5-20% compared to the state-of-the-art methods developed for deterministic imputation.
- We demonstrate that CSDI can also be applied to time series interpolations and probabilistic forecasting, and is competitive with existing baselines designed for these tasks.

2 Related works

Time series imputations with deep learning Previous studies have shown deep learning models can capture the temporal dependency of time series and give more accurate imputation than statistical methods. A popular approach using deep learning is to use RNNs, including LSTMs and GRUs, for sequence modeling [39, 4, 3]. Subsequent studies combined RNNs with other methods to improve imputation performance, such as GANs [14, 15, 17] and self-training [6]. Among them, the combination of RNNs with attention mechanisms is particularly successful for imputation and interpolation of time series [34, 28]. While these methods focused on deterministic imputation, GP-VAE [8] has been recently developed as a probabilistic imputation method.

Score-based generative models Score-based generative models, including score matching with Langevin dynamics [31] and denoising diffusion probabilistic models [9], have outperformed existing methods with other deep generative models in many domains, such as images [31, 9], audio [12, 5], and graphs [21]. Most recently, TimeGrad [23] utilized diffusion probabilistic models for probabilistic time series forecasting. While the method has shown state-of-the-art performance, it cannot be applied to time series imputation due to the use of RNNs to handle past time series.

3 Background

3.1 Multivariate time series imputation

We consider N *multivariate* time series with missing values. Let us denote the values of each time series as $\mathbf{X} = \{x_{1:K,1:L}\} \in \mathbb{R}^{K \times L}$ where K is the number of features and L is the length of time series. While the length L can be different for each time series, we treat the length of all time series as the same for simplicity, unless otherwise stated. We also denote an observation mask as $\mathbf{M} = \{m_{1:K,1:L}\} \in \{0,1\}^{K \times L}$ where $m_{k,l} = 0$ if $x_{k,l}$ is missing, and $m_{k,l} = 1$ if $x_{k,l}$ is observed. We assume time intervals between two consecutive data entries can be different, and define the timestamps of the time series as $\mathbf{s} = \{s_{1:L}\} \in \mathbb{R}^L$. In summary, each time series is expressed as $\{\mathbf{X}, \mathbf{M}, \mathbf{s}\}$.

Probabilistic time series imputation is the task of estimating the distribution of the missing values of $\mathbf{X}$ by exploiting the observed values of $\mathbf{X}$. We note that this definition of imputation includes other related tasks, such as interpolation, which imputes all features at target time points, and forecasting, which imputes all features at future time points.

3.2 Denoising diffusion probabilistic models

Let us consider learning a model distribution $p_\theta(\mathbf{x}_0)$ that approximates a data distribution $q(\mathbf{x}_0)$. Let $\mathbf{x}_t$ for $t = 1, \dots, T$ be a sequence of latent variables in the same sample space as $\mathbf{x}_0$, which is denoted as $\mathcal{X}$. Diffusion probabilistic models [30] are latent variable models that are composed of two processes: the forward process and the reverse process. The forward process adds noise to data $\mathbf{x}_0$. The latent variable $\mathbf{x}_t$ can be expressed as $\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_0 + (1-\alpha_t)\boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\{\alpha_{1:T}\}$ is a sequence of noise levels, which satisfies $1 > \alpha_1 > \cdots > \alpha_T > 0$. The reverse process denoises $\mathbf{x}_t$ to recover $\mathbf{x}_0$, and is defined by the following Markov chain:

$$
\begin{aligned}
p_\theta(\mathbf{x}_{0:T}) &:= p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t), \quad \mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \\
p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) &:= \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \sigma_\theta(\mathbf{x}_t, t)\mathbf{I}).
\end{aligned} \tag{1}
$$

Ho et al. [9] has recently proposed denoising diffusion probabilistic models (DDPM), which considers the following specific parameterization of $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$:

$$
\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \boldsymbol{\mu}^{\text{DDPM}}(\mathbf{x}_t, t, \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)), \quad \sigma_\theta(\mathbf{x}_t, t) = \sigma^{\text{DDPM}}(\mathbf{x}_t, t) \tag{2}
$$

where $\boldsymbol{\mu}^{\text{DDPM}}$ and σ^{DDPM} are non trainable functions (defined in Appendix A) and $\boldsymbol{\epsilon}_\theta : \mathcal{X} \times \mathbb{R} \to \mathcal{X}$ is a trainable denoising function. The denoising function in Eq. (2) also corresponds to a rescaled score model for score-based generative models [31]. Under this parameterization, Ho et al. [9] have shown that the reverse process can be trained by solving the following optimization problem:

$$
\min_\theta \mathcal{L}(\theta) := \min_\theta \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), t} ||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)||_2^2 \quad \text{where } \mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_0 + (1-\alpha_t)\boldsymbol{\epsilon}. \tag{3}
$$

The denoising function $\boldsymbol{\epsilon}_\theta$ estimates the noise vector $\boldsymbol{\epsilon}$ that was added to its noisy input $\mathbf{x}_t$. This training objective also be viewed as a weighted combination of denoising score matching used for training score-based generative models [31, 32, 33]. Once trained, we can sample $\mathbf{x}_0$ from Eq. (1). We provide the details of DDPM in Appendix A.

3.3 Imputation with diffusion models

Here, we focus on general imputation tasks that are not restricted to time series imputation. Let us consider the following imputation problem: given a sample $\mathbf{x}_0$ which contains missing values, we generate imputation targets $\mathbf{x}_0^{\text{ta}} \in \mathcal{X}^{\text{ta}}$ by exploiting conditional observations $\mathbf{x}_0^{\text{co}} \in \mathcal{X}^{\text{co}}$, where $\mathcal{X}^{\text{ta}}$ and $\mathcal{X}^{\text{co}}$ are a part of the sample space $\mathcal{X}$ and vary per sample. Then, the goal of probabilistic imputation is to estimate the true conditional data distribution $q(\mathbf{x}_0^{\text{ta}} \mid \mathbf{x}_0^{\text{co}})$ with a model distribution $p_\theta(\mathbf{x}_0^{\text{ta}} \mid \mathbf{x}_0^{\text{co}})$. We typically impute all missing values using all observed values, and set all observed values as $\mathbf{x}_0^{\text{co}}$ and all missing values as $\mathbf{x}_0^{\text{ta}}$, respectively. Note that time series imputation in Section 3.1 can be considered as a special case of this task.

Let us consider modeling $p_\theta(\mathbf{x}_0^{\mathrm{ta}} \mid \mathbf{x}_0^{\mathrm{co}})$ with a diffusion model. In the unconditional case, the reverse process $p_\theta(\mathbf{x}_{0:T})$ is used to define the final data model $p_\theta(\mathbf{x}_0)$. Then, a natural approach is to extend the reverse process in Eq. (1) to a conditional one:

$$\begin{aligned} p_\theta(\mathbf{x}_{0:T}^{\mathrm{ta}} \mid \mathbf{x}_0^{\mathrm{co}}) &:= p(\mathbf{x}_T^{\mathrm{ta}}) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}^{\mathrm{ta}} \mid \mathbf{x}_t^{\mathrm{ta}}, \mathbf{x}_0^{\mathrm{co}}), \quad \mathbf{x}_T^{\mathrm{ta}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \\ p_\theta(\mathbf{x}_{t-1}^{\mathrm{ta}} \mid \mathbf{x}_t^{\mathrm{ta}}, \mathbf{x}_0^{\mathrm{co}}) &:= \mathcal{N}(\mathbf{x}_{t-1}^{\mathrm{ta}}; \boldsymbol{\mu}_\theta(\mathbf{x}_t^{\mathrm{ta}}, t \mid \mathbf{x}_0^{\mathrm{co}}), \sigma_\theta(\mathbf{x}_t^{\mathrm{ta}}, t \mid \mathbf{x}_0^{\mathrm{co}})\mathbf{I}). \end{aligned} \tag{4}$$

However, existing diffusion models are generally designed for data generation and do not take conditional observations $\mathbf{x}_0^{\mathrm{co}}$ as inputs. To utilize diffusion models for imputation, previous studies [33, 11] approximated the conditional reverse process $p_\theta(\mathbf{x}_{t-1}^{\mathrm{ta}} \mid \mathbf{x}_t^{\mathrm{ta}}, \mathbf{x}_0^{\mathrm{co}})$ with the reverse process in Eq. (1). With this approximation, in the reverse process they add noise to both the target and the conditional observations $\mathbf{x}_0^{\mathrm{co}}$. While this approach can impute missing values, the added noise can harm useful information in the observations. This suggests that modeling $p_\theta(\mathbf{x}_{t-1}^{\mathrm{ta}} \mid \mathbf{x}_t^{\mathrm{ta}}, \mathbf{x}_0^{\mathrm{co}})$ without approximations can improve the imputation quality. Hereafter, we call the model defined in Section 3.2 as the unconditional diffusion model.

4 Conditional score-based diffusion model for imputation (CSDI)

In this section, we propose CSDI, a novel imputation method based on a conditional score-based diffusion model. The conditional diffusion model allows us to exploit useful information in observed values for accurate imputation. We provide the reverse process of the conditional diffusion model, and then develop a self-supervised training method. We note that CSDI is not restricted to time series.

4.1 Imputation with CSDI

We focus on the conditional diffusion model with the reverse process in Eq. (4) and aim to model the conditional distribution $p(\mathbf{x}_{t-1}^{\mathrm{ta}} \mid \mathbf{x}_t^{\mathrm{ta}}, \mathbf{x}_0^{\mathrm{co}})$ without approximations. Specifically, we extend the parameterization of DDPM in Eq. (2) to the conditional case. We define a conditional denoising function $\boldsymbol{\epsilon}_\theta : (\mathcal{X}^{\mathrm{ta}} \times \mathbb{R} \mid \mathcal{X}^{\mathrm{co}}) \to \mathcal{X}^{\mathrm{ta}}$, which takes conditional observations $\mathbf{x}_0^{\mathrm{co}}$ as inputs. Then, we consider the following parameterization with $\boldsymbol{\epsilon}_\theta$:

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t^{\mathrm{ta}}, t \mid \mathbf{x}_0^{\mathrm{co}}) = \boldsymbol{\mu}^{\mathrm{DDPM}}(\mathbf{x}_t^{\mathrm{ta}}, t, \boldsymbol{\epsilon}_\theta(\mathbf{x}_t^{\mathrm{ta}}, t \mid \mathbf{x}_0^{\mathrm{co}})), \quad \sigma_\theta(\mathbf{x}_t^{\mathrm{ta}}, t \mid \mathbf{x}_0^{\mathrm{co}}) = \sigma^{\mathrm{DDPM}}(\mathbf{x}_t^{\mathrm{ta}}, t) \tag{5}$$

where $\boldsymbol{\mu}^{\mathrm{DDPM}}$ and σ^{DDPM} are the same functions as in Eq. (2). Given the function $\boldsymbol{\epsilon}_\theta$ and data $\mathbf{x}_0$, we can sample $\mathbf{x}_0^{\mathrm{ta}}$ using the reverse process in Eq. (4) and Eq. (5). For the sampling, we set all observed values of $\mathbf{x}_0$ as conditional observations $\mathbf{x}_0^{\mathrm{co}}$ and all missing values as imputation targets $\mathbf{x}_0^{\mathrm{ta}}$. Note that the conditional model is reduced to the unconditional one under no conditional observations and can also be used for data generation.

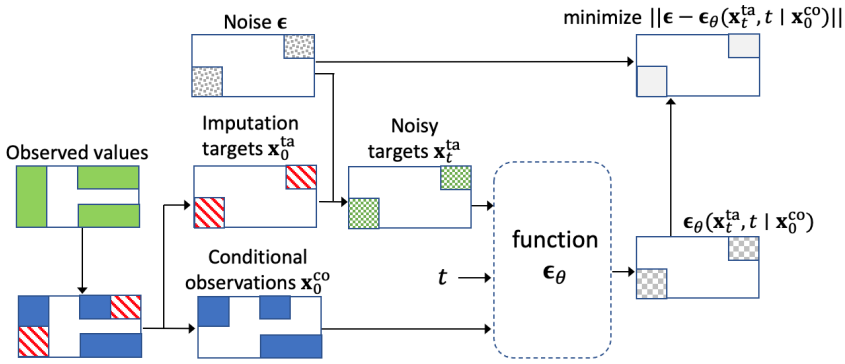

Figure 2: The self-supervised training procedure of CSDI. On the middle left rectangle, the green and white areas represent observed and missing values, respectively. The observed values are separated into red imputation targets $\mathbf{x}_0^{\mathrm{ta}}$ and blue conditional observations $\mathbf{x}_0^{\mathrm{co}}$, and used for training of $\boldsymbol{\epsilon}_\theta$. The colored areas in each rectangle mean the existence of values.

Table 1: Imputation targets $\mathbf{x}_0^{\mathrm{ta}}$ and conditional observations $\mathbf{x}_0^{\mathrm{co}}$ for CSDI at training and sampling.

	imputation targets $\mathbf{x}_0^{\mathrm{ta}}$	conditional observations $\mathbf{x}_0^{\mathrm{co}}$
sampling (imputation)	all missing values	all observed values
training	a subset of the observed values (sampled by a target choice strategy)	the remaining observed values

4.2 Training of CSDI

Since Eq. (5) uses the same parameterization as Eq. (2) and the difference between Eq. (2) and Eq. (5) is only the form of $\boldsymbol{\epsilon}_\theta$, we can follow the training procedure for the unconditional model in Section 3.2. Namely, given conditional observations $\mathbf{x}_0^{\mathrm{co}}$ and imputation targets $\mathbf{x}_0^{\mathrm{ta}}$, we sample noisy targets $\mathbf{x}_t^{\mathrm{ta}} = \sqrt{\alpha_t}\mathbf{x}_0^{\mathrm{ta}} + (1-\alpha_t)\boldsymbol{\epsilon}$, and train $\boldsymbol{\epsilon}_\theta$ by minimizing the following loss function:

$$\min_\theta \mathcal{L}(\theta) := \min_\theta \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0},\mathbf{I}), t} ||(\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t^{\mathrm{ta}}, t \mid \mathbf{x}_0^{\mathrm{co}}))||_2^2 \tag{6}$$

where the dimension of $\boldsymbol{\epsilon}$ corresponds to that of the imputation targets $\mathbf{x}_0^{\mathrm{ta}}$.

However, this training procedure has an issue. Since we do not know the ground-truth missing values in practice, it is not clear how to select $\mathbf{x}_0^{\mathrm{co}}$ and $\mathbf{x}_0^{\mathrm{ta}}$ from a training sample $\mathbf{x}_0$. To address this issue, we develop a self-supervised learning method inspired by masked language modeling [7]. We illustrate the training procedure in Figure 2. Given a sample $\mathbf{x}_0$, we separate observed values of $\mathbf{x}_0$ into two parts, and set one of them as imputation targets $\mathbf{x}_0^{\mathrm{ta}}$ and the other as conditional observations $\mathbf{x}_0^{\mathrm{co}}$. We choose the targets $\mathbf{x}_0^{\mathrm{ta}}$ through a target choice strategy, which is discussed in Section 4.3. Then, we sample noisy targets $\mathbf{x}_t^{\mathrm{ta}}$ and train $\boldsymbol{\epsilon}_\theta$ by solving Eq. (6). We summarize how we set $\mathbf{x}_0^{\mathrm{co}}$ and $\mathbf{x}_0^{\mathrm{ta}}$ for training and sampling in Table 1. We also provide the algorithm of training and sampling in Appendix B.1.

4.3 Choice of imputation targets in self-supervised learning

In the proposed self-supervised learning, the choice of imputation targets is important. We provide four target choice strategies depending on what is known about the missing patterns in the test dataset. We describe the algorithm for these strategies in Appendix B.2.

(1) *Random* strategy : this strategy is used when we do not know about missing patterns, and randomly chooses a certain percentage of observed values as imputation targets. The percentage is sampled from $[0\%, 100\%]$ to adapt to various missing ratios in the test dataset.

(2) *Historical* strategy: this strategy exploits missing patterns in the training dataset. Given a training sample $\mathbf{x}_0$, we randomly draw another sample $\tilde{\mathbf{x}}_0$ from the training dataset. Then, we set the intersection of the observed indices of $\mathbf{x}_0$ and the missing indices of $\tilde{\mathbf{x}}_0$ as imputation targets. The motivation of this strategy comes from structured missing patterns in the real world. For example, missing values often appear consecutively in time series data. When missing patterns in the training and test dataset are highly correlated, this strategy helps the model learn a good conditional distribution.

(3) *Mix* strategy: this strategy is the mix of the above two strategies. The historical strategy may lead to overfitting to missing patterns in the training dataset. The *Mix* strategy can benefit from generalization by the random strategy and structured missing patterns by the historical strategy.

(4) *Test* pattern strategy: when we know the missing patterns in the test dataset, we just set the patterns as imputation targets. For example, this strategy is used for time series forecasting, since the missing patterns in the test dataset are fixed to given future time points.

5 Implementation of CSDI for time series imputation

In this section, we implement CSDI for time series imputation. For the implementation, we need the inputs and the architecture of $\boldsymbol{\epsilon}_\theta$.

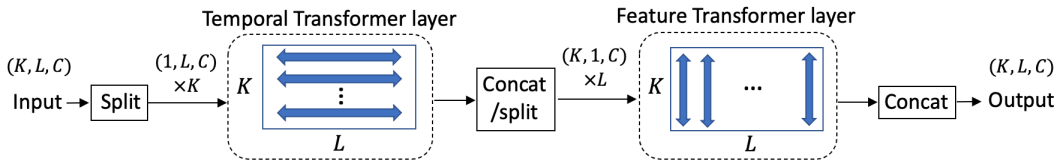

Figure 3: The architecture of 2D attention. Given a tensor with K features, L length, and C channels, the temporal Transformer layer takes tensors with $(1, L, C)$ shape as inputs and learns temporal dependency. The feature Transformer layer takes tensors with $(K, 1, C)$ shape as inputs and learns feature dependency. The output shape of each layer is the same as the input shape.

First, we describe how we process time series data as inputs for CSDI. As defined in Section 3.1, a time series is denoted as $\{\mathbf{X}, \mathbf{M}, \mathbf{s}\}$, and the sample space $\mathcal{X}$ of $\mathbf{X}$ is $\mathbb{R}^{K \times L}$. We want to handle $\mathbf{X}$ in the sample space $\mathbb{R}^{K \times L}$ for learning dependencies in a time series using a neural network, but the conditional denoising function $\boldsymbol{\epsilon}_\theta$ takes inputs $\mathbf{x}_t^{\mathrm{ta}}$ and $\mathbf{x}_0^{\mathrm{co}}$ in varying sample spaces that are a part of $\mathcal{X}$. To address this issue, we adjust the conditional denoising function $\boldsymbol{\epsilon}_\theta$ to inputs in the fixed sample space $\mathbb{R}^{K \times L}$. Concretely, we fix the shape of the inputs $\mathbf{x}_t^{\mathrm{ta}}$ and $\mathbf{x}_0^{\mathrm{co}}$ to $(K \times L)$ by applying zero padding to $\mathbf{x}_t^{\mathrm{ta}}$ and $\mathbf{x}_0^{\mathrm{co}}$. To indicate which indices are padded, we introduce the conditional mask $\mathbf{m}^{\mathrm{co}} \in \{0, 1\}^{K \times L}$ as an additional input to $\boldsymbol{\epsilon}_\theta$, which corresponds to $\mathbf{x}_0^{\mathrm{co}}$ and takes value 1 for indices of conditional observations. Then, $\boldsymbol{\epsilon}_\theta$ is written as $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t^{\mathrm{ta}}, t \mid \mathbf{x}_0^{\mathrm{co}}, \mathbf{m}^{\mathrm{co}})$ where $\mathbf{x}_t^{\mathrm{ta}}, \mathbf{x}_0^{\mathrm{co}} \in \mathbb{R}^{K \times L}$. We discuss the effect of this adjustment on training and sampling in Appendix D.

Under the adjustment, we set conditional observations $\mathbf{x}_0^{\mathrm{co}}$ and imputation targets $\mathbf{x}_0^{\mathrm{ta}}$ for time series imputation by following Table 1. At sampling time, since conditional observations $\mathbf{x}_0^{\mathrm{co}}$ are all observed values, we set $\mathbf{m}^{\mathrm{co}} = \mathbf{M}$ and $\mathbf{x}_0^{\mathrm{co}} = \mathbf{m}^{\mathrm{co}} \odot \mathbf{X}$ where $\odot$ represents element-wise products. For training, we sample $\mathbf{x}_0^{\mathrm{ta}}$ and $\mathbf{x}_0^{\mathrm{co}}$ through a target choice strategy, and set the indices of $\mathbf{x}_0^{\mathrm{co}}$ as $\mathbf{m}^{\mathrm{co}}$. Then, $\mathbf{x}_0^{\mathrm{co}}$ is written as $\mathbf{x}_0^{\mathrm{co}} = \mathbf{m}^{\mathrm{co}} \odot \mathbf{X}$ and $\mathbf{x}_0^{\mathrm{ta}}$ is obtained as $\mathbf{x}_0^{\mathrm{ta}} = (\mathbf{M} - \mathbf{m}^{\mathrm{co}}) \odot \mathbf{X}$.

Next, we describe the architecture of $\boldsymbol{\epsilon}_\theta$. We adopt the architecture in DiffWave [12] as the base, which is composed of multiple residual layers with residual channel C. We refine this architecture for time series imputation. We set the diffusion step $T = 50$. We discuss the main differences from DiffWave (see Appendix E.1 for the whole architecture and details).

Attention mechanism To capture temporal and feature dependencies of multivariate time series, we utilize a two dimensional attention mechanism in each residual layer instead of a convolution architecture. As shown in Figure 3, we introduce temporal Transformer layer and a feature Transformer layer, which are 1-layer Transformer encoders. The temporal Transformer layer takes tensors for each feature as inputs to learn temporal dependency, whereas the feature Transformer layer takes tensors for each time point as inputs to learn temporal dependency.

Note that while the length L can be different for each time series as mentioned in Section 3.1, the attention mechanism allows the model to handle various lengths. For batch training, we apply zero padding to each sequence so that the lengths of the sequences are the same.

Side information In addition to the arguments of $\boldsymbol{\epsilon}_\theta$, we provide some side information as additional inputs to the model. First, we use time embedding of $\mathbf{s} = \{s_{1:L}\}$ to learn the temporal dependency. Following previous studies [36, 40], we use 128-dimensions temporal embedding. Second, we exploit categorical feature embedding for K features. We set the embedding dimension as 16.

6 Experimental results

In this section, we demonstrate the effectiveness of CSDI for time series imputation. Since CSDI can be applied to other related tasks such as interpolation and forecasting, we also evaluate CSDI for these tasks to show the flexibility of CSDI. Due to the page limitation, we provide the detailed setup for experiments including train/validation/test splits and hyperparameters in Appendix E.2.

6.1 Time series imputation

Dataset and experiment settings We run experiments for two datasets. The first one is the healthcare dataset in PhysioNet Challenge 2012 [29], which consists of 4000 clinical time series with 35 variables for 48 hours from intensive care unit (ICU). Following previous studies [3, 4], we process the dataset to hourly time series with 48 time steps. The processed dataset contains around 80% missing values. Since the dataset has no ground-truth, we randomly choose 10/50/90% of observed values as ground-truth on the test data.

The second one is the air quality dataset [38]. Following previous studies [3, 34], we use hourly sampled PM2.5 measurements from 36 stations in Beijing for 12 months and set 36 consecutive time steps as one time series. There are around 13% missing values and the missing patterns are not random. The dataset contains artificial ground-truth, whose missing patterns are also structured.

For both dataset, we run each experiment five times. As the target choice strategy for training, we adopt the random strategy for the healthcare dataset and the mix of the random and historical strategy for the air quality dataset, based on the missing patterns of each dataset.

Results of probabilistic imputation CSDI is compared with two baselines. 1) GP-VAE [8]: the method showed the state-of-the-art results for probabilistic imputation. 2) V-RIN [18]: a deterministic imputation method that uses the uncertainty quantified by VAE to improve imputation. For V-RIN, we regard the quantified uncertainty as probabilistic imputation. We use their original implementations for experiments. In addition, we compare CSDI with imputation using the unconditional diffusion model in order to show the effectiveness of the conditional one (see Appendix C for training and imputation with the unconditional diffusion model).

We first show quantitative results. We adopt the continuous ranked probability score (CRPS) [16] as the metric, which is freuquently used for evaluating probabilistic time series forecasting and measures the compatibility of an estimated probability distribution with an observation. We generate 100 samples to approximate the probability distribution over missing values and report the normalized average of CRPS for all missing values following previous studies [26] (see Appendix E.3 for details of the computation).

Table 2: Comparing CRPS for probabilistic imputation baselines and CSDI (lower is better). We report the mean and the standard error of CRPS for five trials.

	healthcare			air quality
	10% missing	50% missing	90% missing	
GP-VAE [8]	0.574(0.003)	0.774(0.004)	0.998(0.001)	0.397(0.009)
V-RIN [18]	0.808(0.008)	0.831(0.005)	0.922(0.003)	0.526(0.025)
unconditional	0.360(0.007)	0.458(0.008)	0.671(0.007)	0.135(0.001)
CSDI (proposed)	**0.238(0.001)**	**0.330(0.002)**	**0.522(0.002)**	**0.113(0.001)**

Table 2 represents CRPS for each method. CSDI reduces CRPS by 40-70% compared to the existing baselines for both datasets. This indicates that CSDI generates more realistic distributions than other methods. We also observe that the imputation with CSDI outperforms that with the unconditional model. This suggests CSDI benefits from explicitly modeling the conditional distribution.

We provide imputation examples in Figure 4. For the air quality dataset, CSDI (green solid line) provides accurate imputations with high confidence, while those by GP-VAE (gray dashed line) are far from ground-truth. CSDI also gives reasonable imputations for the healthcare dataset. These results indicate that CSDI exploits temporal and feature dependencies to provide accurate imputations. We give more examples in Appendix G.

Results of deterministic imputation We demonstrate that CSDI also provides accurate deterministic imputations, which are obtained as the median of 100 generated samples. We compare CSDI with four baselines developed for deterministic imputation including GLIMA [34], which combined recurrent imputations with an attention mechanism to capture temporal and feature dependencies and

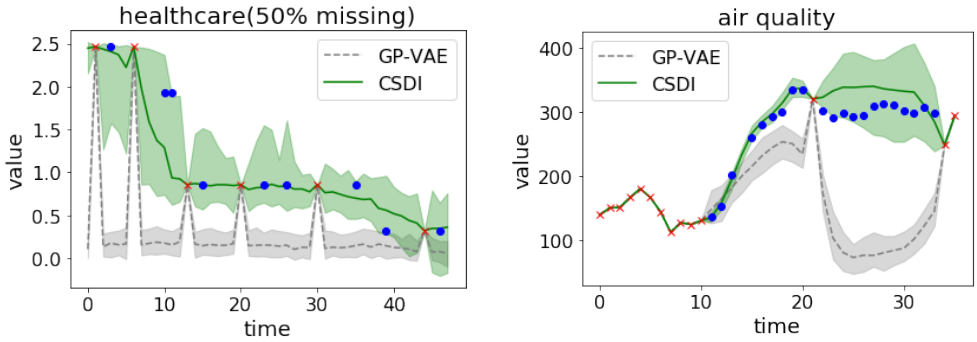

Figure 4: Examples of probabilistic time series imputation for the healthcare dataset with 50% missing (left) and the air quality dataset (right). The red crosses show the observed values and the blue circles show the ground-truth imputation targets. For each method, median values of imputations are shown as the line and 5% and 95% quantiles are shown as the shade.

Table 3: Comparing MAE for deterministic imputation methods and CSDI. We report the mean and the standard error for five trials. The asterisks mean the results of the method are cited from the original paper.

	healthcare			air quality
	10% missing	50% missing	90% missing	
V-RIN [18]	0.271(0.001)	0.365(0.002)	0.606(0.006)	25.4(0.62)
BRITS [3]	0.284(0.001)	0.368(0.002)	0.517(0.002)	14.11(0.26)
BRITS [3] (*)	0.278	–	–	11.56
GLIMA [34] (*)	0.265	–	–	10.54
RDIS [6]	0.319(0.002)	0.419(0.002)	0.631(0.002)	22.11(0.35)
unconditional	0.326(0.008)	0.417(0.010)	0.625(0.010)	12.13(0.07)
CSDI (proposed)	**0.217(0.001)**	**0.301(0.002)**	**0.481(0.003)**	**9.60(0.04)**

showed the state-of-the-art performance. These methods are based on autoregressive models. We use the original implementations except RDIS.

We evaluate each method by the mean absolute error (MAE). In Table 3, CSDI improves MAE by 5-20% compared to the baselines. This suggests that the conditional diffusion model is effective to learn temporal and feature dependencies for imputation. For the healthcare dataset, the gap between the baselines and CSDI is particularly significant when the missing ratio is small, because more observed values help CSDI capture dependencies.

Table 4: Comparing the state-of-the-art interpolation method with CSDI for the healthcare dataset. We report the mean and the standard error of MAE for five trials.

	10% missing	50% missing	90% missing
mTANs [28]	0.401(0.003)	0.424(0.002)	0.539(0.002)
CSDI (proposed)	**0.362(0.001)**	**0.394(0.002)**	**0.518(0.003)**

6.2 Interpolation of irregularly sampled time series

Dataset and experiment settings We use the same healthcare dataset as the previous section, but process the dataset as irregularly sampled time series, following previous studies [28, 25]. Since the dataset has no ground-truth, we randomly choose 10/50/90% of *time* points and use observed values at these time points as ground-truth on the test data. As the target choice strategy for training, we adopt the random strategy, which is adjusted for interpolation so that some *time* points are sampled.

Results We compare CSDI with mTANs [28], which utilized an attention mechanism and showed state-of-the-art results for the interpolation of irregularly sampled time series. We describe the result in Table 4. CSDI outperforms the baseline for all cases. The gap between the baseline and CSDI is significant when the missing ratio is small, which is consistent with the result in Section 6.1.

Table 5: Comparing probabilistic forecasting methods with CSDI. We report the mean and the standard error of CRPS-sum for three trials. The baseline results are cited from the original paper. 'TransMAF' is the abbreviation for 'Transformer MAF'.

	solar	electricity	traffic	taxi	wiki
GP-copula [26]	$0.337(0.024)$	$0.024(0.002)$	$0.078(0.002)$	$0.208(0.183)$	$0.086(0.004)$
TransMAF [24]	$0.301(0.014)$	$0.021(0.000)$	$0.056(0.001)$	$0.179(0.002)$	$0.063(0.003)$
TLAE [20]	$\mathbf{0.124(0.033)}$	$0.040(0.002)$	$0.069(0.001)$	$0.130(0.006)$	$0.241(0.001)$
TimeGrad [23]	$0.287(0.020)$	$0.021(0.001)$	$0.044(0.006)$	$\mathbf{0.114(0.020)}$	$0.049(0.002)$
CSDI (proposed)	$0.298(0.004)$	$\mathbf{0.017(0.000)}$	$\mathbf{0.020(0.001)}$	$0.123(0.003)$	$\mathbf{0.047(0.003)}$

6.3 Time series Forecasting

Dataset and Experiment settings We use five datasets that are commonly used for evaluating probabilistic time series forecasting. Each dataset is composed of around 100 to 2000 features. We predict all features at future time steps using past time series. We use the same prediction steps as previous studies [26, 20]. For the target choice strategy, we adopt the *Test* pattern strategy.

Results We compare CSDI with four baselines. Specifically, TimeGrad [23] combined the diffusion model with a RNN-based encoder. We evaluate each method for CRPS-sum, which is CRPS for the distribution of the sum of all time series across K features and accounts for joint effect (see Appendix E.3 for details).

In Table 5, CSDI outperforms the baselines for electricity and traffic datasets, and is competitive with the baselines as a whole. The advantage of CSDI over baselines for forecasting is smaller than that for imputation in Section 6.1. We hypothesize it is because the datasets for forecasting seldom contains missing values and are suitable for existing encoders including RNNs. For imputation, it is relatively difficult for RNNs to handle time series due to missing values.

7 Conclusion

In this paper, we have proposed CSDI, a novel approach to impute multivariate time series with conditional diffusion models. We have shown that CSDI outperforms the existing probabilistic and deterministic imputation methods.

There are some interesting directions for future work. One direction is to improve the computation efficiency. While diffusion models generate plausible samples, sampling is generally slower than other generative models. To mitigate the issue, a recent study [33] utilized ODE solver for fast sampling. Another study [12] also proposed a fast sampling algorithm. Combining our method with these approaches would likely improve the sampling efficiency.

Another direction is to extend CSDI to downstream tasks such as classifications. Many previous studies have shown that accurate imputation improves the performance on downstream tasks [3, 15, 28]. Since conditional diffusion models can learn temporal and feature dependencies with uncertainty, joint training of imputations and downstream tasks using conditional diffusion models would be helpful to improve the performance of the downstream tasks.

Finally, although our focus was on time series, it would be interesting to explore CSDI as imputation technique on other modalities.

References

[1] Alexander Alexandrov, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C. Maddix, Syama Rangapuram, David Salinas, Jasper

Schulz, Lorenzo Stella, Ali Caner Türkmen, and Yuyang Wang. GluonTS: Probabilistic and Neural Time Series Modeling in Python. *Journal of Machine Learning Research*, 21(116):1–6, 2020.

[2] S van Buuren and Karin Groothuis-Oudshoorn. MICE: Multivariate imputation by chained equations in r. *Journal of statistical software*, pages 1–68, 2010.

[3] Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. BRITS: Bidirectional recurrent imputation for time series. In *Advances in Neural Information Processing Systems*, 2018.

[4] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):1–12, 2018.

[5] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. WaveGrad: Estimating gradients for waveform generation. In *International Conference on Learning Representations*, 2021.

[6] Tae-Min Choi, Ji-Su Kang, and Jong-Hwan Kim. RDIS: Random drop imputation with self-training for incomplete time series data. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence*, 2021.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[8] Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch, and Stephan Mandt. GP-VAE: Deep probabilistic time series imputation. In *International Conference on Artificial Intelligence and Statistics*, 2020.

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

[10] Andrew T Hudak, Nicholas L Crookston, Jeffrey S Evans, David E Hall, and Michael J Falkowski. Nearest neighbor imputation of species-level, plot-scale forest structure attributes from lidar data. *Remote Sensing of Environment*, 112(5):2232–2245, 2008.

[11] Zahra Kadkhodaie and Eero P Simoncelli. Solving linear inverse problems using the prior implicit in a denoiser. *arXiv preprint arXiv:2007.13640*, 2020.

[12] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. DiffWave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*, 2021.

[13] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 95–104, 2018.

[14] Yonghong Luo, Xiangrui Cai, Ying Zhang, Jun Xu, and Xiaojie Yuan. Multivariate time series imputation with generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 1603–1614, 2018.

[15] Yonghong Luo, Ying Zhang, Xiangrui Cai, and Xiaojie Yuan. E2GAN: End-to-end generative adversarial network for multivariate time series imputation. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 3094–3100, 2019.

[16] James E Matheson and Robert L Winkler. Scoring rules for continuous probability distributions. *Management science*, 22(10):1087–1096, 1976.

[17] Xiaoye Miao, Yangyang Wu, Jun Wang, Yunjun Gao, Xudong Mao, and Jianwei Yin. Generative semi-supervised learning for multivariate time series imputation. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence*, 2021.

[18] Ahmad Wisnu Mulyadi, Eunji Jun, and Heung-Il Suk. Uncertainty-aware variational-recurrent imputation network for clinical time series. *IEEE Transactions on Cybernetics*, 2021. to appear.

[19] Fulufhelo V Nelwamondo, Shakir Mohamed, and Tshilidzi Marwala. Missing data: A comparison of neural network and expectation maximization techniques. *Current Science*, pages 1514–1521, 2007.

[20] Nam Nguyen and Brian Quanz. Temporal latent auto-encoder: A method for probabilistic multivariate time series forecasting. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence*, 2021.

[21] Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling. In *International Conference on Artificial Intelligence and Statistics*, pages 4474–4484. PMLR, 2020.

[22] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 2019.

[23] Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. *arXiv preprint arXiv:2101.12072*, 2021.

[24] Kashif Rasul, Abdul-Saboor Sheikh, Ingmar Schuster, Urs Bergmann, and Roland Vollgraf. Multi-variate probabilistic time series forecasting via conditioned normalizing flows. In *International Conference on Learning Representations*, 2021.

[25] Yulia Rubanova, Ricky TQ Chen, and David Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems*, 2019.

[26] David Salinas, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus. High-dimensional multivariate forecasting with low-rank gaussian copula processes. In *Advances in Neural Information Processing Systems*, 2019.

[27] Zhuoran Shen, Mingyuan Zhang, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Efficient attention: Attention with linear complexities. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3531–3539, 2021.

[28] Satya Narayan Shukla and Benjamin M Marlin. Multi-time attention networks for irregularly sampled time series. In *International Conference on Learning Representations*, 2021.

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

[30] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, 2015.

[31] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, 2019.

[32] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. In *Advances in Neural Information Processing Systems*, 2020.

[33] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[34] Qiuling Suo, Weida Zhong, Guangxu Xun, Jianhui Sun, Changyou Chen, and Aidong Zhang. 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, 2020.

[35] Huachun Tan, Guangdong Feng, Jianshuai Feng, Wuhong Wang, Yu-Jin Zhang, and Feng Li. A tensor-based method for missing traffic data completion. *Transportation Research Part C: Emerging Technologies*, 28:15–27, 2013.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

[37] Phil Wang. Linear attention transformer. `https://github.com/lucidrains/linear-attention-transformer`, 2020.

[38] Xiuwen Yi, Yu Zheng, Junbo Zhang, and Tianrui Li. ST-MVL: filling missing values in geo-sensory time series data. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 2704–2710, 2016.

[39] 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.

[40] Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. Transformer hawkes process. In *International Conference on Machine Learning*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] We described our proposed method and empirical results in the abstract and introduction.
   (b) Did you describe the limitations of your work? [Yes] We have mentioned some limitations of our work in Section 7.
   (c) Did you discuss any potential negative societal impacts of your work? [No] Since time series imputation is not a particularly sensitive topic, we did not discuss negative societal impacts in the main paper.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] We have read the guidelines and ensured that our paper conforms to them.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We attached the code in the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 6 and Appendix E.2.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We gave error bars in figures and the standard errors in tables.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] We did not focus on computational time.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cited packages we used. We also cited papers which provided datasets.
   (b) Did you mention the license of the assets? [No] We only used open dataset.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No] We did not release new assets.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] We only used open dataset.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] The datasets we used are open numerical time series dataset and do not contain personally identifiable information.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]