Probabilistic Imputation for Time-series Classification with Missing Data

SeungHyun Kim [* 1] **Hyunsu Kim** [* 1] **EungGu Yun** [* 1 2] **Hwangrae Lee** [3] **Jaehun Lee** [3] **Juho Lee** [1 4]

Abstract

Multivariate time series data for real-world applications typically contain a significant amount of missing values. The dominant approach for classification with such missing values is to impute them heuristically with specific values (zero, mean, values of adjacent time-steps) or learnable parameters. However, these simple strategies do not take the data generative process into account, and more importantly, do not effectively capture the uncertainty in prediction due to the multiple possibilities for the missing values. In this paper, we propose a novel probabilistic framework for classification with multivariate time series data with missing values. Our model consists of two parts; a deep generative model for missing value imputation and a classifier. Extending the existing deep generative models to better capture structures of time-series data, our deep generative model part is trained to impute the missing values in multiple plausible ways, effectively modeling the uncertainty of the imputation. The classifier part takes the time series data along with the imputed missing values and classifies signals, and is trained to capture the predictive uncertainty due to the multiple possibilities of imputations. Importantly, we show that naïvely combining the generative model and the classifier could result in trivial solutions where the generative model does not produce meaningful imputations. To resolve this, we present a novel regularization technique that can promote the model to produce useful imputation values that help classification. Through extensive experiments on real-world time series data with missing values, we demonstrate the effectiveness of our method.

[*]Equal contribution [1]Kim Jaechul Graduate School of AI, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea [2]Saige Research, Seoul, South Korea [3]Samsung Research, Seoul, South Korea [4]AITRICS, Seoul, South Korea. Correspondence to: SeungHyun Kim <sh0776879@gmail.com>, Juho Lee <juholee@kaist.ac.kr>.

1. Introduction

Multivariate time-series data are universal; many real-world applications ranging from healthcare, stock markets, and weather forecasting take multivariate time-series data as inputs. Arguably the biggest challenge in dealing with such data is the presence of missing values, due to the fundamental difficulty of faithfully measuring data for all time steps. The degree of missing is often severe, so in some applications, more than 90% of data are missing for some features. Therefore, developing an algorithm that can accurately and robustly perform predictions with missing data is considered an important problem to be tackled.

In this paper, we focus on the task of classification, where the primary goal is to classify given multivariate time-series data with missing values, simply imputing the missing values with heuristically chosen values considered to be strong baselines that are often competitive or even better than more sophisticated methods. For instance, one can fill all the missing values with zero, the mean of the data, or values from the previous time steps. GRU-D (Che et al., 2018) proposes a more elaborated imputation algorithm where the missing values are filled with a mixture between the data means and values from the previous time steps with the mixing coefficients learned from the data. While these simple imputation-based methods work surprisingly well (Che et al., 2018; Du et al., 2022), they lack a fundamental mechanism to recover the missing values, especially the underlying generative process of the given time series data.

Dealing with missing data is deeply connected to handling uncertainties originating from the fact that there may be multiple plausible options for filling in the missing values, so it is natural to analyze them with the probabilistic framework. There have been rich literature on statistical analysis for missing data, where the primary goal is to understand how the observed and missing data are generated. In the seminal work of Little & Rubin (2002), three assumptions for the missing data generative process were introduced, including Missing Completely At Random (MCAR), Missing At Random (MAR), and Missing Not At Random (MNAR). While MCAR or MAR simplifies the modeling and thus makes the inference easier, they may be unrealistic for real-world applications, because they assume that the missing mechanism is independent of the missing values (MAR) or

both missing and observed values (MCAR). MNAR, the most generic assumption, assumes that the missing mechanism depends on both missing and observed values, so the generative model based on the MNAR assumption should explicitly take the missing mechanism into account. Based on this framework, Mattei & Frellsen (2019) presented deep generative models for missing data under MAR assumption, and this was later extended to MNAR in Ipsen et al. (2021). Combining the deep generative model and classifier, Ipsen et al. (2022) proposed a hybrid model that can classify missing data with problematically imputed values generated under MAR assumption.

Still, in our opinion, there is no satisfactory work combining probabilistic generative models for multivariate time-series data with missing values and classification models, so that the classifier could consider the uncertainty in filling in the missing values when making predictions. The aforementioned probabilistic frameworks are not designed for classification (Mattei & Frellsen, 2019; Ipsen et al., 2021), and more importantly, not tailored for time series data (Ipsen et al., 2022). A naïve extension of Ipsen et al. (2022) for time series is likely to fail; putting the obvious difference between the static and time series data aside, the fundamental difficulty of learning the generative models for missing is that there are no explicit learning signals that could promote the model to generate "meaningful" missing values. Since we don't have ground truth for the missing values, in principle, the generative model can generate arbitrary values (e.g., zeros), and the combined classifier can still successfully classify time series data, which is a critical problem that is overlooked in the existing works.

To this end, we propose a hybrid model combining the deep generative models for multivariate time series data and the classification models for them. The generative part is built under the MNAR assumption and is designed to naturally encode the continuity of the multivariate time series data. The classifier then takes the missing values generated from the generative model to classify time-series, and unlike the algorithms based on heuristic imputations, it takes multiple feasible options for the missing values and computes predictions based on them. To tackle the difficulty in guiding the generative model to generate "meaningful" missing values, we introduce a novel regularization technique that deliberately erases *observed values* during training. As a consequence, the classifier is forced to do classification based more on the generated missing values, so the generative model is encouraged to produce missing values that are more advantageous for the classification. Using the various real-world multivariate time series benchmarks with missing values, we demonstrate that our approach outperforms baselines both in terms of classification accuracy and uncertainty estimates.

2. Background

2.1. Settings and Notations

Let $\boldsymbol{x} = [x_1, \ldots, x_d]^\top \in \mathbb{R}^d$ be a d-dimensional vector, along with the mask vector $\boldsymbol{s} = [s_1, \ldots, s_d]^\top \in \{0, 1\}^d$, where $s_j = 1$ if x_j is observed and $s_j = 0$ otherwise. Given a mask $\boldsymbol{s}$, we can split $\boldsymbol{x}$ into the observed part $\boldsymbol{x}^{\mathrm{o}} := \{x_j \mid s_j = 1\}$ and the missing part $\boldsymbol{x}^{\mathrm{m}} := \{x_j \mid s_j = 0\}$. For a collection of data, the ith instance is denoted as $\boldsymbol{x}_i = [x_{i,1}, \ldots, x_{i,d}]$, and $\boldsymbol{s}_i$, $\boldsymbol{x}_i^{\mathrm{o}}$, and $\boldsymbol{x}_i^{\mathrm{m}}$ are defined similarly. For a multivariate time-series data, we denote the vector of tth time step as $\boldsymbol{x}_t = [x_{t,1}, \ldots, x_{t,d}] \in \mathbb{R}^d$, and the corresponding mask as $\boldsymbol{s}_t = [s_{t,1}, \ldots, s_{t,d}]$. The tth time step of ith instance of a collection is denoted as $\boldsymbol{x}_{t,i}$, which is split into $\boldsymbol{x}_{t,i}^{\mathrm{o}}$ and $\boldsymbol{x}_{t,i}^{\mathrm{m}}$ according to $\boldsymbol{s}_{t,i}$.

Following Mattei & Frellsen (2019); Ipsen et al. (2021), we assume that the joint distribution of an input $\boldsymbol{x}$ and a mask $\boldsymbol{s}$ is factorized as $p_{\boldsymbol{\theta}, \boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{s}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}) p_{\boldsymbol{\psi}}(\boldsymbol{s} | \boldsymbol{x})$. The conditional distribution $p_{\boldsymbol{\psi}}(\boldsymbol{s} | \boldsymbol{x})$ plays an important role for describing missing mechanism. Under MCAR assumption, we have $p(\boldsymbol{s} | \boldsymbol{x}) = p(\boldsymbol{s})$, under MAR we have $p_{\boldsymbol{\psi}}(\boldsymbol{s} | \boldsymbol{x}) = p_{\boldsymbol{\psi}}(\boldsymbol{s} | \boldsymbol{x}^{\mathrm{o}})$, and under MNAR we have $p_{\boldsymbol{\psi}}(\boldsymbol{s} | \boldsymbol{x}) = p_{\boldsymbol{\psi}}(\boldsymbol{s} | \boldsymbol{x}^{\mathrm{o}}, \boldsymbol{x}^{\mathrm{m}})$. The likelihood for the observed data $\boldsymbol{x}^{\mathrm{o}}$ is thus computed as $p_{\boldsymbol{\theta}, \boldsymbol{\psi}}(\boldsymbol{x}^{\mathrm{o}}, \boldsymbol{s}) = \int p_{\boldsymbol{\theta}, \boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{s}) \mathrm{d}\boldsymbol{x}^{\mathrm{m}}$.

2.2. Missing Data Importance-Weighted Autoencoder and its extensions

In this section, we briefly review the Missing data Importance-Weighted AutoEncoder (MIWAE) (Mattei & Frellsen, 2019), a deep generative model for missing data, and its extensions to MNAR and supervised settings. Similar to variational autoencoder (VAE) (Kingma & Welling, 2014), MIWAE assumes that a data $\boldsymbol{x}$ is genearted from a latent representation $\boldsymbol{z}$, but we only observe $\boldsymbol{x}^{\mathrm{o}}$ with $\boldsymbol{s}$ generated from the missing model $p_{\boldsymbol{\psi}}(\boldsymbol{s} | \boldsymbol{x})$. MIWAE assumes MAR, so we have $p_{\boldsymbol{\psi}}(\boldsymbol{s} | \boldsymbol{x}) = p_{\boldsymbol{\psi}}(\boldsymbol{s} | \boldsymbol{x}^{\mathrm{o}})$. The log-likelihood for $(\boldsymbol{x}^{\mathrm{o}}, \boldsymbol{s})$ is then computed as

$$\log p_{\boldsymbol{\theta}, \boldsymbol{\psi}}(\boldsymbol{x}^{\mathrm{o}}, \boldsymbol{s})$$
$$= \log p_{\boldsymbol{\psi}}(\boldsymbol{s} | \boldsymbol{x}^{\mathrm{o}}) + \underbrace{\log \int p_{\boldsymbol{\theta}}(\boldsymbol{x}^{\mathrm{o}} | \boldsymbol{z}) p_{\boldsymbol{\theta}}(\boldsymbol{z}) \mathrm{d}\boldsymbol{z}}_{= \log p_{\boldsymbol{\theta}}(\boldsymbol{x}^{\mathrm{o}})} . \quad (1)$$

For the missing data imputation, $p_{\boldsymbol{\psi}}(\boldsymbol{s} | \boldsymbol{x}^{\mathrm{o}})$ is not necessary, so we choose to maximize only the $\log p_{\boldsymbol{\theta}}(\boldsymbol{x}^{\mathrm{o}})$. The integral is intractable, so we consider the Importance Weighted AutoEncoder (IWAE) lower bound (Burda et al., 2015) as a proxy loss,

$$\mathcal{L}_{\mathrm{MIWAE}}^{(K)}(\boldsymbol{\theta}, \boldsymbol{\phi}) := \mathbb{E}\left[\log \frac{1}{K} \sum_{k=1}^{K} \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}^{\mathrm{o}} | \boldsymbol{z}_k) p_{\boldsymbol{\theta}}(\boldsymbol{z}_k)}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_k | \boldsymbol{x}^{\mathrm{o}})} \right]. \quad (2)$$

Here, $q_{\phi}(z_k|x^{\mathrm{o}})$ for $k = 1, \ldots, K$ are i.i.d. copies of the variational distribution (encoder) $q_{\phi}(z|x^{\mathrm{o}})$ approximating the true posterior $p_{\phi}(z|x^{\mathrm{o}})$, and the expectation is w.r.t. $\prod_{k=1}^{K} q(z_k|x^{\mathrm{o}})$. $\mathbb{E}_{z_{1:K}}$ denotes the expectation w.r.t. $\prod_{k=1}^{K} q_{\phi}(z_k|x^{\mathrm{o}})$. K is the number of particles, and the bound converges to the log-likelihood as $K \to \infty$, that is, $\mathcal{L}_{\mathrm{MIWAE}}^{(1)}(\theta, \phi) \leq \mathcal{L}_{\mathrm{MIWAE}}^{(2)}(\theta, \phi) \leq \cdots = \log p_{\theta}(x^{\mathrm{o}})$.

Ipsen et al. (2021) presented not-MIWAE, an extension of MIWAE with MNAR assumption. The log-likelihood for (x^{o}, s) under the MNAR assumption is,

$$
\log p_{\theta,\psi}(x^{\mathrm{o}}, s) = \log \int p_{\psi}(s|x^{\mathrm{o}}, x^{\mathrm{m}}) p_{\theta}(x^{\mathrm{o}}|z) \\
\times p_{\theta}(x^{\mathrm{m}}|z) p_{\theta}(z) \mathrm{d}z \mathrm{d}x^{\mathrm{m}}, \quad (3)
$$

where we assume that $(x^{\mathrm{o}}, x^{\mathrm{m}})$ are independent given z. The corresponding IWAE lower-bound with the variational distribution $q_{\phi}(x^{\mathrm{m}}, z|x^{\mathrm{o}}) = p_{\theta}(x^{\mathrm{m}}|z) q_{\phi}(z|x^{\mathrm{o}})$ is,

$$
\mathcal{L}_{\mathrm{notMIWAE}}^{(K)}(\theta, \psi, \phi) \\
:= \mathbb{E}\left[\log \frac{1}{K} \sum_{k=1}^{K} \frac{p_{\theta}(s|x^{\mathrm{o}}, x_k^{\mathrm{m}}) p_{\theta}(x^{\mathrm{o}}|z_k) p_{\theta}(z_k)}{q_{\phi}(z_k|x^{\mathrm{o}})} \right], \quad (4)
$$

where the expectation is w.r.t. $\prod_{k=1}^{K} p_{\theta}(x_k^{\mathrm{m}}|z_k) q_{\phi}(z_k|x^{\mathrm{o}})$.

On the other hand, Ipsen et al. (2022) extended MIWAE to a supervised learning setting, where the goal is to learn the joint distribution of an observed input x^{o}, a mask s, and corresponding label y,

$$
\log p_{\theta,\psi,\lambda}(y, x^{\mathrm{o}}, s) = \log p_{\psi}(s|x^{\mathrm{o}}) \\
+ \underbrace{\log \int p_{\lambda}(y|x^{\mathrm{o}}, x^{\mathrm{m}}) p_{\theta}(x^{\mathrm{o}}|z) p_{\theta}(x^{\mathrm{m}}|z) p_{\theta}(z) \mathrm{d}z}_{=\log p_{\theta,\lambda}(y, x^{\mathrm{o}})}, \quad (5)
$$

The term $p_{\psi}(s|x^{\mathrm{o}})$ is irrelevant to the prediction for y, so we choose to maximize $\log p_{\theta,\lambda}(y, x^{\mathrm{o}})$, which again can be lower-bounded by IWAE bound with the variational distribution $q_{\phi}(z, x^{\mathrm{m}}|x^{\mathrm{o}}) = p_{\theta}(x^{\mathrm{m}}|z) q_{\phi}(z|x^{\mathrm{o}})$:

$$
\mathcal{L}_{\mathrm{supMIWAE}}^{(K)}(\theta, \lambda, \phi) \\
:= \mathbb{E}\left[\log \frac{1}{K} \sum_{k=1}^{K} \frac{p_{\lambda}(y|x^{\mathrm{o}}, x_k^{\mathrm{m}}) p_{\theta}(x^{\mathrm{o}}|z_k) p(z_k)}{q_{\phi}(z_k|x^{\mathrm{o}})} \right], \quad (6)
$$

where the expectation is w.r.t. $\prod_{k=1}^{K} p_{\theta}(x_k^{\mathrm{m}}|z_k) q_{\phi}(z_k|x^{\mathrm{o}})$.

2.3. GRU for multivariate time series data and imputation methods

We briefly review GRU (Cho et al., 2014) and its variant for time series classification with missing data since they are common baselines. Given a multivariate time series

$(x_t)_{t=1}^{T}$, GRU takes a vector of one time step at a time and accumulates the information into a hidden state h_t. Specifically, the forward pass at t^{th} time step takes x_t and updates the hidden state h_t as follows:

$$
a_t = \sigma(W_a x_t + U_a h_{t-1} + b_a), \\
r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \\
\tilde{h}_t = \tanh(W x_t + U(r_t \odot h_{t-1}) + b), \\
h_t = (1 - a_t) \odot h_{t-1} + a_t \odot \tilde{h}_t,
$$

where $\odot$ denotes the element-wise multiplication. We also review the heuristical imputation methods described in Che et al. (2018), which are common baselines for the related methods. Let $\hat{x}_{t,j}$ denote the imputed value for $x_{t,j}$.

- **GRU-zero**: a zero padding setting $\hat{x}_{t,j} = s_{t,j} x_{t,j}$.

- **GRU-mean**: imputes the missing values as $\hat{x}_{t,j} = s_{t,j} x_{t,j} + (1 - s_{t,j}) \bar{x}_j$, where $\bar{x}_j = \sum_{i=1}^{n} \sum_{t=1}^{T} s_{t,i,j} x_{t,i,j} / \sum_{i=1}^{n} \sum_{t=1}^{T} s_{t,i,j}$ is the empirical mean of observed values for j^{th} feature of a given collection of time series data $((x_{t,i})_{t=1}^{T})_{i=1}^{n}$.

- **GRU-forward**: set $\hat{x}_{t,j} = s_{t,j} x_{t,j} + (1 - s_{t,j}) x_{t',j}$, where t' is the last time when j^{th} feature was observed before t.

- **GRU-simple**: along with the imputed vector $\hat{x}_t$ (either by GRU-mean or GRU-forward), concatenate additional information. Che et al. (2018) proposed to concatenate 1) the mask s_t, and the *time-interval* δ_t saving the length of the intervals between observed values (see Che et al. (2018) for precise definition). The concatenated vector $[\hat{x}_t, s_t, \delta_t]$ is then fed into GRU.

- **GRU-D**: introduces *learnable decay* values for the input x_t and hidden state h_t as follows:

$$
\gamma_{x_t} = \exp(-\max(W_{\gamma_x} \delta_t + b_{\gamma_x}, 0)), \\
\gamma_{h_t} = \exp(-\max(W_{\gamma_h} \delta_t + b_{\gamma_h}, 0)).
$$

Given a vector x_t with mask s_t, GRU-D imputes the missing values as

$$
\hat{x}_{t,j} = \mathrm{Decay}(s_{t,j}, x_{t,j}, \gamma_{x_t,j}, x_{t',j}, \bar{x}_j) \\
:= s_{t,j} x_{t,j} + (1 - s_{t,j})(\gamma_{x_t,j} x_{t',j} \\
+ (1 - \gamma_{x_t,j}) \bar{x}_j). \quad (7)
$$

That is, the missing is imputed as a mixture of the last observed $x_{t',j}$ and the empirical mean $\bar{x}_j$ with the mixing coefficient set as the learned decay. The hidden state from the previous time step h_{t-1} is decayed as $\gamma_{h_t} \odot h_{t-1}$ and passed through GRU along with the imputed $\hat{x}_t$.

3. Methods

In this section, we describe our method, a probabilistic framework for multivariate time series data with missing values. Our method is an extension of supMIWAE to time series data under MNAR assumption, but the actual implementation is not merely a naïve composition of the existing models. In Section 3.1, we first present supnot-MIWAE, an MNAR version of supMIWAE, with the encoder and decoder architectures designed for time series data with missing values. In Section 3.2, we show why the sup(not)MIWAE for data with missings may fail, and propose a novel regularization technique to prevent that.

3.1. supnotMIWAE for multivariate time series data

Given a multivariate time series data $\boldsymbol{x}_{1:T} := (\boldsymbol{x}_t)_{t=1}^T$ with observed $\boldsymbol{x}_{1:T}^{\mathrm{o}}$, missing $\boldsymbol{x}_{1:T}^{\mathrm{m}}$, a missing mask $\boldsymbol{s}_{1:T} := (\boldsymbol{s}_t)_{t=1}^T$, and a label $\boldsymbol{y}$, we assume the following state-space model with latent vectors $\boldsymbol{z}_{1:T} := (\boldsymbol{z}_t)_{t=1}^T$.

$$
\begin{aligned}
&p_{\boldsymbol{\theta},\boldsymbol{\psi},\boldsymbol{\lambda}}(\boldsymbol{y}, \boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{s}_{1:T}) \\
&= \int p_{\boldsymbol{\lambda}}(\boldsymbol{y}|\boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{x}_{1:T}^{\mathrm{m}}) p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{o}}|\boldsymbol{z}_{1:T}) p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{m}}|\boldsymbol{z}_{1:T}) \\
&\quad \times p_{\boldsymbol{\theta}}(\boldsymbol{z}_{1:T}) p_{\boldsymbol{\psi}}(\boldsymbol{s}_{1:T}|\boldsymbol{x}_{1:T}) \mathrm{d}\boldsymbol{x}_{1:T}^{\mathrm{m}} \mathrm{d}\boldsymbol{z}_{1:T}.
\end{aligned} \tag{8}
$$

Below we describe each component more in detail.

Prior $p_{\boldsymbol{\theta}}(\boldsymbol{z}_{1:T})$. We assume Gaussian process prior as in Fortuin et al. (2020) for $\boldsymbol{z}_{1:T}$ to encode temporal correlation in the latent space. Let $\boldsymbol{z}_{1:T,j} = [z_{1,j}, \ldots, z_{T,j}]^\top$ be the vector collecting j^{th} dimension of the series $\boldsymbol{z}_{1:T}$.

$$
p_{\boldsymbol{\theta}}(\boldsymbol{z}_{1:T}) = \prod_{j=1}^d \mathcal{N}(\boldsymbol{z}_{1:T,j}|\boldsymbol{0}, \boldsymbol{K}), \tag{9}
$$

where $\boldsymbol{K}_{ij} = k(t_i, t_j)$ $i, j \in \{1 \ldots T\}$ and k is kernel function. We use a Cauchy kernel for all experiments in this paper.

Decoders $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{o}}|\boldsymbol{z}_{1:T})$ **and** $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{m}}|\boldsymbol{z}_{1:T})$. The decoder for the observed $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{o}}|\boldsymbol{z}_{1:T})$ is defined in an autoregressive fashion,

$$
\begin{aligned}
&p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{o}}|\boldsymbol{z}_{1:T}) \\
&= \prod_{t=1}^T \mathcal{N}(\boldsymbol{x}_t^{\mathrm{o}}|\boldsymbol{\mu}_{\mathrm{dec}}(\boldsymbol{z}_{1:t}), \mathrm{diag}(\boldsymbol{\sigma}_{\mathrm{dec}}^2(\boldsymbol{z}_{1:t}))),
\end{aligned} \tag{10}
$$

where $(\boldsymbol{\mu}_{\mathrm{dec}}(\boldsymbol{z}_{1:t}), \boldsymbol{\sigma}_{\mathrm{dec}}(\boldsymbol{z}_{1:t}))_{t=1}^T$ are defined with a transformer (Vaswani et al., 2017) with causal maskings.

$$
\begin{aligned}
\boldsymbol{h}_t &= \mathrm{Transformer}_{\mathrm{dec}}(\boldsymbol{z}_{1:t}), \\
(\boldsymbol{\mu}_{\mathrm{dec}}(\boldsymbol{z}_{1:t}), \boldsymbol{\sigma}_{\mathrm{dec}}(\boldsymbol{z}_{1:t})) &= \mathrm{MLP}_{\mathrm{dec}}(\boldsymbol{h}_t).
\end{aligned} \tag{11}
$$

In practice, this casual transformer layer is applied at times. The decoder for the missing $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{m}}|\boldsymbol{z}_{1:T})$ defined similarly. In our implementation, we actually let them share the same model generating both $\boldsymbol{x}_t^{\mathrm{o}}$ and $\boldsymbol{x}_t^{\mathrm{m}}$.

Missing model $p_{\boldsymbol{\psi}}(\boldsymbol{s}_{1:T}|\boldsymbol{x}_{1:T})$. The missing model is simply assumed to be independent Bernoulli distributions over the time steps and features.

$$
p_{\boldsymbol{\psi}}(\boldsymbol{s}_{1:T}|\boldsymbol{x}_{1:T}) = \prod_{t=1}^T \prod_{j=1}^d \mathrm{Bern}(s_{t,j}|\sigma_{\mathrm{mis},t,j}(\boldsymbol{x}_{1:T})), \tag{12}
$$

where $\sigma_{\mathrm{mis}}(\boldsymbol{x}_{1:T})$ is computed as

$$
\sigma_{\mathrm{mis}}(\boldsymbol{x}_{1:T}) = \mathrm{MLP}_{\mathrm{mis}}(\boldsymbol{x}_{1:T}). \tag{13}
$$

Classifier $p_{\boldsymbol{\lambda}}(\boldsymbol{y}|\boldsymbol{x}_{1:T}^{\mathbf{o}}, \boldsymbol{x}_{1:T}^{\mathbf{m}})$ We use a transformer based model for the classifier. Given a time-series data $\boldsymbol{x}_{1:T}$ packing the observed values $\boldsymbol{x}_{1:T}^{\mathrm{o}}$ and the imputed missing values generated from the decoders, we first process the data with 1D CNN applied along the time axis to compute $\boldsymbol{r}_{1:T} := \mathrm{CNN}(\boldsymbol{x}_{1:T})$. Then we process $\boldsymbol{r}_{1:T}$ with a Transformer block to compute an output $\boldsymbol{h}_T$. The conditional distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{y}|\boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{x}_{1:T}^{\mathrm{m}})$ is defined as

$$
\mathrm{Categorical}(\boldsymbol{y} \,|\, \mathrm{Softmax}(\mathrm{Linear}_{\mathrm{cls}}(\boldsymbol{h}_T)). \tag{14}
$$

During the forward pass, the classifier takes the observed input $\boldsymbol{x}_{1:T}^{\mathrm{o}}$ and the missing values *generated* from the decoder $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{m}}|\boldsymbol{z}_{1:T})$. We find it beneficial to adopt the idea of GRU-D, where instead of directly putting the generated missing values $\boldsymbol{x}_{1:T}^{\mathrm{m}}$, putting the *decayed* missing values as follows:

$$
\begin{aligned}
\tilde{\boldsymbol{x}}_{1:T} &:= (\boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{x}_{1:T}^{\mathrm{m}}) \text{ where } \boldsymbol{x}_{1:T}^{\mathrm{m}} \sim p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{m}}|\boldsymbol{z}_{1:T}), \\
\hat{x}_{t,j} &= \mathrm{Decay}(s_{t,j}, x_{t,j}, \gamma_{\hat{\boldsymbol{x}}_t,j}, x_{t',j}, \tilde{x}_{t,j}).
\end{aligned} \tag{15}
$$

where $\boldsymbol{\gamma}_{\hat{\boldsymbol{x}}_t} = \exp(-\max(\boldsymbol{0}, \boldsymbol{W}_{\hat{\boldsymbol{x}}}\boldsymbol{\delta}_t + \boldsymbol{b}_{\hat{\boldsymbol{x}}})$ is a learnable decay. We find this stabilizes the learning when the generated missing values $\boldsymbol{x}_{1:T}^{\mathrm{m}}$ are inaccurate, for instance, in the early stage of learning. Note also the difference between (15) and the original GRU-D imputation (7). In GRU-D, the last observed values are mixed with the mean feature, while ours mix them with the generated values.

Encoder $q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}})$. Given the generative model defined above, we introduce the variational distribution for $(\boldsymbol{x}_{1:T}^{\mathrm{m}}, \boldsymbol{z}_{1:T})$ that factorizes as,

$$
p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{m}}|\boldsymbol{z}_{1:T}) q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}}). \tag{16}
$$

Here, the encoder $q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}})$ is defined as an autoregressive model as before,

$$
\begin{aligned}
&q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}}) \\
&= \prod_{t=1}^T \mathcal{N}(\boldsymbol{z}_t|\boldsymbol{\mu}_{\mathrm{enc}}(\boldsymbol{x}_{1:t}^{\mathrm{o}}), \mathrm{diag}(\boldsymbol{\sigma}_{\mathrm{enc}}^2(\boldsymbol{x}_{1:t}^{\mathrm{o}}))).
\end{aligned} \tag{17}
$$

Given a series of observed values $\boldsymbol{x}_{1:T}^{\mathrm{o}}$, we first apply zero imputation for the missing values, that is, set $x'_{t,j} = x_{t,j}^{\mathrm{o}}$ if $s_{t,j} = 1$ and $x'_{t,j} = 0$ otherwise. Then we concatenate the missing indicators to $\boldsymbol{x}'_{1:T}$ and apply the 1D CNN to the time-axis as $\boldsymbol{r}_{1:T} = \mathrm{CNN}(\boldsymbol{x}'_{1:T})$. Having computed $\boldsymbol{r}_{1:T}$, similar to the decoder, we use a transformer with causal masking to compute

$$\boldsymbol{h}_t = \mathrm{Transformer_{enc}}(\boldsymbol{r}_{1:t}),$$
$$(\boldsymbol{\mu}_{\mathrm{enc}}(\boldsymbol{x}_{1:t}^{\mathrm{o}}), \boldsymbol{\sigma}_{\mathrm{enc}}(\boldsymbol{x}_{1:t}^{\mathrm{o}})) = \mathrm{MLP_{enc}}(\boldsymbol{h}_t). \quad (18)$$

Objective. Having all the ingredients defined, the IWAE bound for supnotMIWAE is computed as follows:

$$\mathcal{L}^{(K)}(\boldsymbol{\lambda}, \boldsymbol{\theta}, \boldsymbol{\psi}, \boldsymbol{\phi}) := \mathbb{E}\left[\log \frac{1}{K}\sum_{k=1}^{K}\omega_k\right], \quad (19)$$

where the expectation is w.r.t. K copies of a variational distribution, $\prod_{k=1}^{K} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{k,1:T}^{\mathrm{m}}|\boldsymbol{z}_{k,1:T})q_{\boldsymbol{\phi}}(\boldsymbol{z}_{k,1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}})$, and ω_k is the importance weight term defined as

$$\omega_k := p_{\boldsymbol{\lambda}}(\boldsymbol{y}|\boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{x}_{k,1:T}^{\mathrm{m}})p_{\boldsymbol{\psi}}(\boldsymbol{s}_{1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{x}_{k,1:T}^{\mathrm{m}})$$
$$\times p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{o}}|\boldsymbol{z}_{k,1:T})p_{\boldsymbol{\theta}}(\boldsymbol{z}_{k,1:T})/q_{\boldsymbol{\phi}}(\boldsymbol{z}_{k,1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}}). \quad (20)$$

3.2. ObsDropout: regularizing supnotMIWAE for better imputation

The problem with (19) is that there is no clear supervision for the missing values $\boldsymbol{x}_{1:T}^{\mathrm{m}}$. Obviously, if we had an access to the missing values, the conditional probability $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{m}}|\boldsymbol{z}_{1:T})$ would guide the model to learn to correctly impute those missing values. Without such true values, we can only encourage the model to impute the missing values with some indirect criteria. In the objective (19), there are two terms that the model hinges on for this matter.

- The missing model $p_{\boldsymbol{\psi}}(\boldsymbol{s}_{1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{x}_{1:T}^{\mathrm{m}})$: this term encourages the model to reconstruct the missing mask s_t from the imputed value x_t^{m}, so in principle, the model should impute the missing values in a way that they are distinguishable from the observed values. However, in general, the distributions of the observed and the missings are not necessarily different, and more importantly, the model can easily cheat the objective. For instance, consider a trivial case where the model imputes all the missing values with zero. The conditional probability $p_{\boldsymbol{\psi}}(\boldsymbol{s}_{1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{x}_{1:T}^{\mathrm{m}})$ can still be maximized by setting $\sigma_{\mathrm{mis}}(x_{t,j}) = 0$ if $x_{t,j} = 0$ (unless there are not many observed with $x_{t,j}^{\mathrm{o}} = 0$).

- The classifier $p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{x}_{1:T}^{\mathrm{m}})$: this term expects the model to generate meaningful imputations so that they are helpful for the classification. However, as shown

in prior works (Che et al., 2018), the classifier can achieve decent classification accuracy *without* meaningful imputations, for instance, it will still be able to classify the signals while all the missing values are imputed with zeros. Hence, in the current form, there is no strong incentive for the model to learn non-trivial imputations that will bring significant accuracy gain over zero imputations.

To summarize, a model trained with the objective (19) is not likely to generate realistic missing values. To resolve this, we may introduce a missing model $p_{\boldsymbol{\theta}}(\boldsymbol{s}_{1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{x}_{1:T}^{\mathrm{m}})$ much more elaborated than the simple i.i.d. model that we are using right now, but that may require some dataset-specific design. Instead, we present a simple regularization technique that can effectively enhance the quality of the imputed values.

Our idea is simple; when passing the observed inputs $\boldsymbol{x}_{1:T}^{\mathrm{o}}$ and the imputed missing values $\hat{\boldsymbol{x}}_{1:T}^{\mathrm{m}}$ (i.e., imputed by (15)) to the classifier, *deliberately drop* some portion of the observed inputs. Without dropping the observed inputs, the classifier may heavily rely on the observed inputs to do the classification, but if some of the observed inputs are dropped out during training, the classifier can focus more on the imputed missing values $\hat{\boldsymbol{x}}_{1:T}^{\mathrm{m}}$. As a result, the model is encouraged to generate more "useful" missing values that are beneficial for classification. More specifically, let β be a predefined dropout probability. Then we construct the imputed input $\hat{\boldsymbol{x}}_t$ to the classifier as follows:

$$\tilde{\boldsymbol{x}}_{1:T} := (\boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{x}_{1:T}^{\mathrm{m}}) \text{ where } \boldsymbol{x}_{1:T}^{\mathrm{m}} \sim p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{m}}|\boldsymbol{z}_{1:T}),$$
$$m_{t,j} \sim \mathrm{Bern}(1 - \beta),$$
$$\hat{x}_{t,j} := \mathrm{Decay}(s_{t,j}m_{t,j}, x_{t,j}, \gamma_{\hat{\boldsymbol{x}}_t,j}, x_{t',j}, \tilde{x}_{t,j}). \quad (21)$$

That is, when an observed $x_{t,j}$ is dropped out, we put a generated value with the decay applied as in (15), so that the classifier could focus more on the values generated by the decoder as we intended. We call this idea *ObsDropout*, since we are dropping out the observed values during the training.

With the mask variables $\boldsymbol{m}_{1:T}$ included, the likelihood is extended

$$p_{\boldsymbol{\theta}, \boldsymbol{\psi}, \boldsymbol{\lambda}}(\boldsymbol{y}, \boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{s}_{1:T})$$
$$= \int p_{\boldsymbol{\lambda}}(\boldsymbol{y}|\boldsymbol{x}_{1:T}^{\mathrm{o}}, \boldsymbol{x}_{1:T}^{\mathrm{m}}, \boldsymbol{m}_{1:T})p_{\beta}(\boldsymbol{m}_{1:T})$$
$$\times p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{o}}|\boldsymbol{z}_{1:T})p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}^{\mathrm{m}}|\boldsymbol{z}_{1:T})$$
$$\times p_{\boldsymbol{\theta}}(\boldsymbol{z}_{1:T})p_{\boldsymbol{\psi}}(\boldsymbol{s}_{1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}})\mathrm{d}\boldsymbol{x}_{1:T}^{\mathrm{m}}\mathrm{d}\boldsymbol{z}_{1:T}\mathrm{d}\boldsymbol{m}_{1:T}. \quad (22)$$

The corresponding IWAE objective is defined similarly to Eq. 19, with the expectation taken with respect to K copies of a variational distribution,
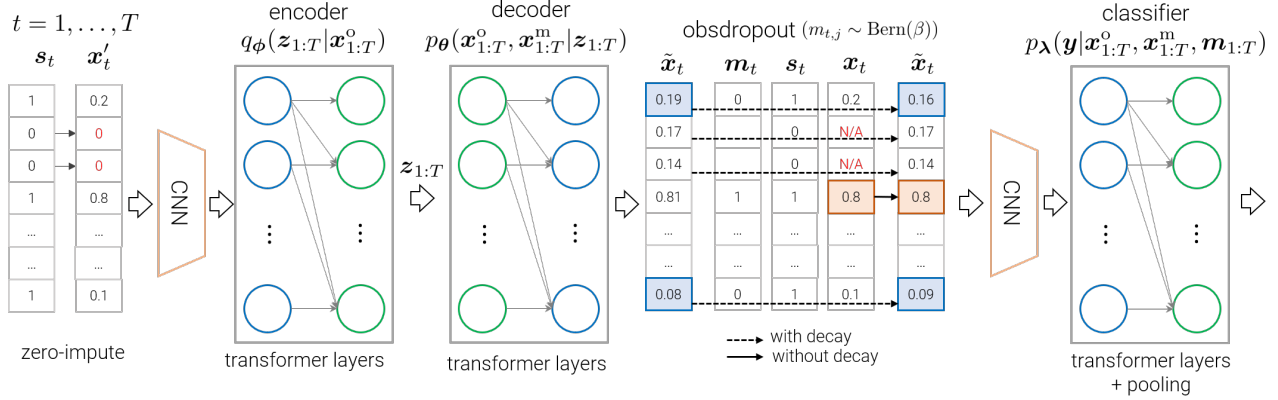
Figure 1. An overview of our model with obsdropout.

$\prod_{k=1}^{K} p_{\theta}(\boldsymbol{x}_{k,1:T}^{\mathrm{m}}|\boldsymbol{z}_{k,1:T})q_{\phi}(\boldsymbol{z}_{k,1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}})p_{\beta}(\boldsymbol{m}_{k,1:T})$ and the importance term is defined as

$$\omega_k := p_{\theta}(\boldsymbol{y}|\boldsymbol{x}_{1:T}^{\mathrm{o}},\boldsymbol{x}_{k,1:T}^{\mathrm{m}},\boldsymbol{m}_{k,1:T})p_{\psi}(\boldsymbol{s}_{1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}},\boldsymbol{x}_{k,1:T}^{\mathrm{m}})$$
$$\times p_{\theta}(\boldsymbol{x}_{1:T}^{\mathrm{o}}|\boldsymbol{z}_{k,1:T})p_{\theta}(\boldsymbol{z}_{k,1:T})/q_{\phi}(\boldsymbol{z}_{k,1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}}), \quad (23)$$

where $p_{\theta}(\boldsymbol{m}_{1:T}) := \prod_{t=1}^{T}\prod_{j=1}^{d}\mathrm{Bern}(m_{t,j}|\beta)$.

3.3. Prediction

Similar to SupMIWAE, we exploit Self-Normalized Importance Sampling (SNIS) to approximate the predictive distribution for a new input $\boldsymbol{x}_{1:T}^{\mathrm{o}}$. With the model trained with obsdropout, we have

$$p(\boldsymbol{y}|\boldsymbol{x}_{1:T}^{\mathrm{o}}) \approx \frac{1}{S}\sum_{s=1}^{S}\sum_{k=1}^{K}\bar{\zeta}_{k}^{(s)}, \quad (24)$$

where

$$(\boldsymbol{z}_{k,1:T}^{(s)}, (\boldsymbol{x}^{\mathrm{m}})_{k,1:T}^{(s)}, \boldsymbol{m}_{k,1:T}^{(s)})$$
$$\overset{\text{i.i.d.}}{\sim} q_{\phi}(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}^{\mathrm{o}})p_{\theta}(\boldsymbol{x}_{1:T}^{\mathrm{m}}|\boldsymbol{z}_{1:T})p_{\beta}(\boldsymbol{m}_{1:T}), \quad (25)$$
$$\zeta_{k}^{(s)} := p_{\lambda}(\boldsymbol{y}|\boldsymbol{x}_{1:T}^{\mathrm{o}}, (\boldsymbol{x}^{\mathrm{m}})_{k,1:T}^{(s)}, \boldsymbol{m}_{k,1:T}^{(s)})p_{\theta}(\boldsymbol{x}_{1:T}^{\mathrm{o}}|\boldsymbol{z}_{k,1:T}^{(s)})$$
$$\times p_{\theta}(\boldsymbol{z}_{k,1:T}^{(s)})/q_{\phi}(\boldsymbol{z}_{k,1:T}^{(s)}|\boldsymbol{x}_{1:T}^{\mathrm{o}}), \quad (26)$$
$$\bar{\zeta}_{k}^{(s)} := \zeta_{k}^{(s)}/\sum_{\ell=1}^{k}\zeta_{\ell}^{(s)}. \quad (27)$$

4. Related Work

There are two lines of literature closely related to our method. The first line consists of work dealing with the problem of missing data imputation based on Deep Latent Variable Models (DLVMs). The other line consists of work designing a tailored neural network architecture for supervised learning on irregularly sampled time series data, which also implies missingness in the process of handling it as a tensor.

DLVMs for missing data Mattei & Frellsen (2019) proposed the MIWAE bound for training DLVMs in the presence of missing data under the MAR assumption. Ipsen et al. (2021) modified the MIWAE bound suitable for the MNAR scenario. Ipsen et al. (2022) extended the MIWAE bound to the supervised learning task. This line of work provides a useful framework for training DLVMs under missingness. However, it is not directly applicable to time series data because it cannot model the temporal dependency within a series. There is previous work that makes DLVMs suitable for multivariate time series. For example, Fortuin et al. (2020) proposed a CNN-based VAE architecture with a Gaussian Process prior to encode the temporal correlation in the latent space. Rubanova et al. (2019) presented ODE-RNNs, which employ Neural Ordinary Differential Equations (Neural ODEs) to model hidden state dynamics of RNNs. Shukla & Marlin (2021a) developed an attention-based VAE architecture with probabilistic interpolation for irregularly sampled time series data.

Irregularly sampled time series classification Researchers have developed deep neural network architectures customized to classify irregularly sampled time series data. Several architectures have shown competitive empirical performance in this task. Che et al. (2018) modified the architecture of GRU intending to perform supervised learning with sparse covariates by introducing a learnable temporal decay mechanism for the input and hidden state of GRU. This mechanism has been applied to further research. For example, Cao et al. (2018) employed temporal decay in hidden states of their bidirectional-RNN-based model to capture the missing pattern of irregularly sampled times series. Shukla & Marlin (2019) presented a hybrid architecture of an interpolation network and a classifier. The interpolation network takes irregularly sampled time series as input and returns fully observed and regularly sampled representation of the original time series data. Shukla &

Table 1. Classification performances of baseline methods and ours on MIMIC-III dataset.

Method	AUROC (↑)	CE (↓)	ECE (↓)	BS (↓)
GRU-Mean	0.844 ± 0.001	0.477 ± 0.012	0.240 ± 0.010	0.154 ± 0.004
GRU-Simple	0.824 ± 0.003	0.451 ± 0.053	0.197 ± 0.039	0.150 ± 0.020
GRU-Forward	0.856 ± 0.002	0.455 ± 0.014	0.219 ± 0.012	0.149 ± 0.005
GRU-D	0.855 ± 0.001	0.468 ± 0.011	0.224 ± 0.009	0.157 ± 0.004
PhasedLSTM	0.802 ± 0.005	0.499 ± 0.049	0.229 ± 0.033	0.166 ± 0.019
IP-Nets	0.830 ± 0.005	0.509 ± 0.031	0.253 ± 0.020	0.172 ± 0.012
SeFT	0.837 ± 0.003	$\mathbf{0.432} \pm 0.013$	$\mathbf{0.186} \pm 0.008$	0.142 ± 0.005
Ours	$\mathbf{0.860} \pm 0.003$	$\mathbf{0.432} \pm 0.016$	0.197 ± 0.011	$\mathbf{0.141} \pm 0.008$
w/o obsdropout	0.849 ± 0.003	0.490 ± 0.024	0.232 ± 0.015	0.161 ± 0.009
w/o MNAR	0.851 ± 0.001	0.462 ± 0.007	0.220 ± 0.007	0.151 ± 0.003

Table 2. Classification performances of baseline methods and ours on PhysioNet 2012 dataset.

Method	AUROC (↑)	CE (↓)	ECE (↓)	BS (↓)
GRU-Mean	0.853 ± 0.003	0.439 ± 0.013	0.194 ± 0.013	0.144 ± 0.006
GRU-Simple	0.819 ± 0.005	0.457 ± 0.065	0.145 ± 0.058	0.150 ± 0.026
GRU-Forward	0.850 ± 0.006	0.475 ± 0.028	0.221 ± 0.026	0.156 ± 0.010
GRU-D	0.862 ± 0.003	0.416 ± 0.024	0.166 ± 0.025	0.135 ± 0.009
PhasedLSTM	0.802 ± 0.007	0.489 ± 0.089	0.199 ± 0.074	0.166 ± 0.035
IP-Nets	0.863 ± 0.003	0.424 ± 0.041	0.180 ± 0.043	0.138 ± 0.015
SeFT	0.863 ± 0.001	0.458 ± 0.044	0.209 ± 0.039	0.156 ± 0.015
Ours	$\mathbf{0.874} \pm 0.004$	0.399 ± 0.057	0.154 ± 0.048	0.129 ± 0.021
w/o obsdropout	0.866 ± 0.004	$\mathbf{0.371} \pm 0.050$	$\mathbf{0.119} \pm 0.041$	$\mathbf{0.116} \pm 0.018$
w/o MNAR	0.869 ± 0.001	0.429 ± 0.033	0.182 ± 0.019	0.140 ± 0.012

Marlin (2021b) later modified interpolation network with an attention-based architecture.

5. Experiments

In this section, we demonstrate our method on real-world multivariate time series data with missing values. We compare ours to the baselines on three datasets: MIMIC-III (Johnson et al., 2016), PhysioNet 2012 (Silva et al., 2012), and Human Activity Recognition (Anguita et al., 2013). MIMIC-III and PhysioNet 2012 datasets contain Electronic Health Records of patients from Intensive Care Units (ICU). Human Activity Recognition dataset consists of the 3D coordinate of sensors mounted on the people doing some daily activities such as walking and sitting. See Appendix A for the details of datasets. For all three datasets, we compare classification accuracy and the uncertainty quantification performances in Section 5.1. We also compare the missing value imputation performance of our methods to the baselines in Section 5.2

For the baselines, we considered GRU classifiers with various imputation methods and other deep neural network based methods that are considered to be competitive in the literature. See Appendix A for the detailed description of the baselines. For the uncertainty quantification metrics, we compared cross-entropy (CE, equals negative log-likelihood), expected calibration error (ECE), and brier score (BS). Please refer to Appendix A for the detailed description of the metrics.

5.1. Classification Results

We summarize the classification results in Table 1, Table 2, and Table 3. In general, our method achieves the best performance among the competing methods both in terms of prediction accuracy and uncertainty quantification. In all classification experiments, our method beats other baseline methods by a wide margin in terms of predictive accuracy. Also, our model shows competitive results with respect to uncertainty metrics. Although SeFT (Horn et al., 2020)

Table 3. Classification performance of baseline methods and ours on Human Activity Recognition dataset.

Method	Accuracy (↑)	CE (↓)	ECE (↓)	BS (↓)
GRU-Mean	0.780 ± 0.005	0.163 ± 0.012	0.019 ± 0.006	0.046 ± 0.002
GRU-Simple	0.861 ± 0.005	0.070 ± 0.004	0.010 ± 0.002	0.019 ± 0.001
GRU-Forward	0.847 ± 0.006	0.084 ± 0.008	0.012 ± 0.003	0.021 ± 0.001
GRU-D	0.860 ± 0.005	0.081 ± 0.005	0.012 ± 0.001	0.020 ± 0.001
PhasedLSTM	0.852 ± 0.001	0.070 ± 0.004	0.008 ± 0.003	0.020 ± 0.001
SeFT	0.848 ± 0.005	0.068 ± 0.001	$\mathbf{0.003} \pm 0.001$	0.020 ± 0.001
Ours	$\mathbf{0.883} \pm 0.003$	$\mathbf{0.063} \pm 0.003$	0.009 ± 0.001	$\mathbf{0.016} \pm 0.000$
w/o obsdropout	0.867 ± 0.018	0.068 ± 0.004	0.008 ± 0.003	0.018 ± 0.000
w/o MNAR	0.882 ± 0.004	0.070 ± 0.004	0.011 ± 0.001	0.017 ± 0.000

shows strong results in terms of uncertainty quantification in some experiments, this method shows inferior performance with respect to predictive accuracy. Since it is reasonable to compare uncertainty quantification between models with similar predictive performance, it can be said that our model shows the best performance among baseline models in general. We also provide an ablation study for our model to see the effect of 1) obsdropout and 2) MNAR assumption. The results clearly show that both components play important roles in our model. For all the experiments, obsdropout clearly makes the gain in terms of predictive performance. Also, removing MNAR assumption makes the worse performance.

5.2. Imputation Results

We quantitatively check the imputation performance of our model on three datasets in Table 4, and visually check the imputation quality by changing our model settings in Fig. 2. Although our model is designed for the classification, ours achieved the lowest MAE and MRE, outperforming the baseline (SAITS) specifically designed for the imputation. Also, our model achieved better performance than GP-VAE model which employs identical prior distribution. This implies that our architecture design and other ingredients such as obsdropout and missing model and supervised signal helps the imputation. Only forward imputation shows comparable performance with our model. However, the structure of forward imputation is very restricted so it cannot deal with various
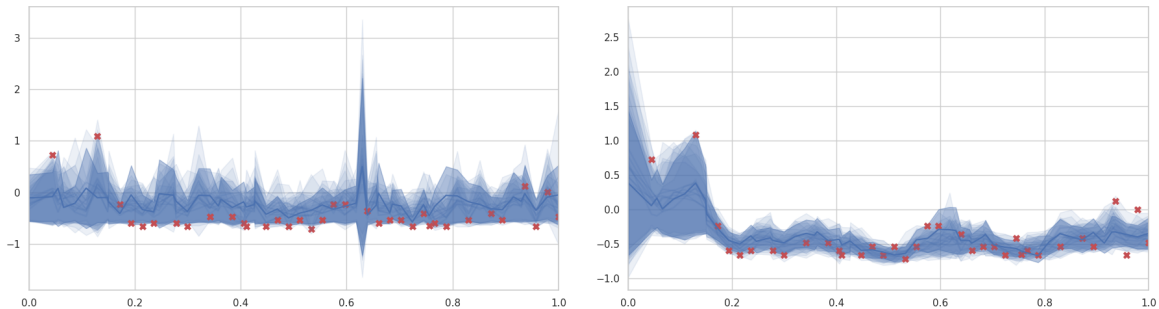
Figure 2. Plots of $\boldsymbol{\mu}_{\text{dec}}(\boldsymbol{z}_{1:t}), \boldsymbol{\sigma}^2_{\text{dec}}(\boldsymbol{z}_{1:t})$. (**Left**) Our model with MLP encoder and MLP decoder. (**Right**) Our model trained with obsdropout with a rate of 0.4. Since MLP architecture does not take the temporal association into account, it shows spiky imputation while our model shows smooth imputation. Also, our model shows better performance in uncertainty quantification since the learned variance of the decoder captures sudden spikes and considers the initial part of the time series more uncertain.

Table 4. Imputation performance on PhysioNet 2012, MIMIC-III and Human Activity Recognition dataset.

Method	PhysioNet 2012		MIMIC-III		Human Activity Recognition	
	MAE ($\downarrow$)	MRE ($\downarrow$)	MAE ($\downarrow$)	MRE ($\downarrow$)	MAE ($\downarrow$)	MRE ($\downarrow$)
Mean	0.696 ± 0.001	1.000 ± 0.000	0.330 ± 0.002	1.000 ± 0.000	0.799 ± 0.001	1.000 ± 0.000
Forward	0.400 ± 0.004	0.576 ± 0.005	0.151 ± 0.002	0.459 ± 0.002	0.305 ± 0.005	$\mathbf{0.373} \pm 0.007$
GP-VAE	0.439 ± 0.010	0.630 ± 0.005	0.198 ± 0.012	0.601 ± 0.031	0.548 ± 0.008	0.684 ± 0.019
SAITS	0.653 ± 0.030	0.942 ± 0.013	0.341 ± 0.005	1.040 ± 0.087	0.834 ± 0.005	1.048 ± 0.013
Ours	$\mathbf{0.367} \pm 0.005$	$\mathbf{0.526} \pm 0.005$	0.149 ± 0.002	0.451 ± 0.008	$\mathbf{0.297} \pm 0.005$	$\mathbf{0.373} \pm 0.006$
w/o supervision	0.376 ± 0.007	0.541 ± 0.009	$\mathbf{0.148} \pm 0.002$	$\mathbf{0.449} \pm 0.006$	0.298 ± 0.005	$\mathbf{0.373} \pm 0.007$
w/o obsdropout	0.377 ± 0.004	0.542 ± 0.005	0.152 ± 0.001	0.459 ± 0.002	0.299 ± 0.005	0.374 ± 0.006
w/o supervision & MNAR	0.394 ± 0.003	0.570 ± 0.006	0.150 ± 0.002	0.457 ± 0.003	0.299 ± 0.005	0.374 ± 0.006

time series such as time series that have many sudden spikes or periodicity. Since our model simultaneously employs the decay mechanism and generative model, our model is more flexible and able to cope with various cases. Especially, the ablation study on the class supervision part $p_\lambda(\boldsymbol{y}|\boldsymbol{x}_{1:T})$ and the obsdropout and MNAR assumption implies that the imputation values generated by our model which was trained to better classify the signals are more "realistic". Fig. 2 highlights the effect of using transformer-based encoders and decoders. The values imputed with those techniques form smoother trajectories and better capture the uncertainties in the intervals without observed values.

6. Conclusion

In this paper, we presented a novel probabilistic framework for multivariate time series classification with missing data. Under the MNAR assumption, we first developed a deep generative model suitable for generating missing values in multivariate time series data. Then we identified an important drawback of the naïve combination of the deep generative models with the classifiers and proposed a novel regularization technique called obsdropout to circumvent that. In this way, combining the MNAR assumption and the

obsdropout regularization technique, the generative model can generate more natural imputation, and the classifier can also perform more accurate and robust classification through this. Also, by using the transformer layers in the internal modules, it can effectively capture the time series structure. Through experiments, we show that it is possible to achieve high performance and uncertainty calibration at the same time in classification tasks with missing values. We demonstrated that ours could classify real-world multivariate time series data more accurately and robustly than existing methods.

Reproducibility statement Please refer to Appendix A for full experimental detail including datasets, models, and evaluation metrics.

Acknowledgement

References

Anguita, D., Ghio, A., Oneto, L., Parra, X., and Reyes-Ortiz, J. L. A public domain dataset for human activity recognition using smartphones. *21th European Symposium on Artificial Neural Networks*, 2013. 7

Burda, Y., Grosse, R., and Salakhutdinov, R. Importance weighted autoencoders. *arXiv preprint arxiv:1509.00519*, 2015. 2

Cao, W., Wang, D., Li, J., Zhou, H., Li, L., and Li, Y. BRITS: bidirectional recurrent imputation for time series. In *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, 2018. 6

Che, Z., Purushotham, S., Cho, K., Sontag, D., and Liu, Y. Recurrent neural networks for multivariate time series with missing values. *Scientific Reports*, 8(1), 2018. 1, 3, 5, 6

Cho, K., van Merriënboer, B., Gulcehre, G., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014. 3

Du, W., Cote, D., and Liu, Y. SAITS: self-attention-based imputation for time series. *arXiv preprint arXiv:2202.08516*, 2022. 1, 13

Fortuin, V., Baranchuk, D., Raetsch, G., and Mandt, S. GP-VAE: Deep probabilistic time series imputation. In *Proceedings of The 23rd International Conference on Artificial Intelligence and Statistics (AISTATS 2020)*, 2020. 4, 6, 13

Harutyunyan, H., Khachatrian, H., Kale, D. C., Ver Steeg, G., and Galstyan, A. Multitask learning and benchmarking with clinical time series data. *Scientific data*, 6(1): 1–18, 2019. 10

Horn, M., Moor, M., Bock, C., Rieck, B., and Borgwardt, K. Set Functions for Time Series. In *Proceedings of The 37th International Conference on Machine Learning (ICML 2020)*, 2020. 7, 10, 11, 12

Ipsen, N. B., Mattei, P., and Frellsen, J. not-MIWAE: deep generative modelling with missing not at random data. In *International Conference on Learning Representations (ICLR)*, 2021. 2, 3, 6

Ipsen, N. B., Mattei, P., and Frellsen, J. How to deal with missing data in supervised deep learning? In *International Conference on Learning Representations (ICLR)*, 2022. 2, 3, 6

Johnson, A., Pollard, T., and Mark, R. MIMIC-III clinical database (version 1.4). 2016. URL https://doi.org/10.13026/C2XW26. 7

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014. 2

Little, R. and Rubin, D. *Statistical analysis with missing data*. Wiley, 2002. 1

Mattei, P. and Frellsen, J. MIWAE: deep generative modelling and imputation of incomplete data sets. In *Proceedings of The 36th International Conference on Machine Learning (ICML 2019)*, 2019. 2, 6

Neil, D., Pfeiffer, M., and Liu, S.-C. Phased lstm: Accelerating recurrent network training for long or event-based sequences. *neurips2016*, 2016. 11

Rubanova, Y., Chen, R. T. Q., and Duvenaud, D. Latent ODEs for Irregularly-Sampled Time Series. In *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, 2019. 6, 10, 11

Shukla, S. N. and Marlin, B. M. Interpolation-Prediction Networks for Irregularly sampled Time Series. In *International Conference on Learning Representations (ICLR)*, 2019. 6, 11

Shukla, S. N. and Marlin, B. M. Heteroscedastic temporal variational autoencoder for irregularly sampled time series. *arXiv preprint arXiv:2107.11350*, 2021a. 6

Shukla, S. N. and Marlin, B. M. Multi-time attention networks for irregularly sampled time series. *arXiv preprint arXiv:2101.10318*, 2021b. 6

Silva, I., Moody, G., Scott, D. J., Celi, L. A., and Mark, R. G. Predicting in-hospital mortality of patients in icu: The physionet/computing in cardiology challenge 2012. *Computing in Cardiology*, 2012. 7

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, 2017. 4

Table 5. Statistics of each dataset.

	PhysioNet2012	MIMIC-III	Human Activity
Number of samples	11,971	21,107	6554
Number of variables	37	16	12
Maximum number of time steps	215	292	50
Mean of number of time steps	74	78	162.4
Total missing rate	0.843	0.655	0.75

A. Experimental Details

A.1. Datasets

A.1.1. DATASET DESCRIPTION

We use three irregularly sampled time series datasets to evaluate the classification and imputation performance of our model and baseline models.

PhysioNet Challenge 2012 (PhysioNet 2012) This dataset contains approximately 12,000 Electronic Health Records of adult patients who were admitted to the intensive care unit (ICU). Each record contains up to 37 time series variables including vital signs such as heart rate, and temperature. All of the variables were measured during the first 48 hours of each patient's admission to ICU and the sampling rate of times series varies among variables. After preprocessing, we have 37 features and 11,971 data points. This data is extremely sparse, with 85% of the entries being unobserved.

MIMIC-III MIMIC-III dataset is a widely used database that includes de-identified Electronical Health Record of patients who stayed in ICU of Beth Israel Deaconess Medical Center from 2001 to 2012. It originally consists of approximately 57,000 records of patients who stayed in ICU. Records include various variables such as medications, in-hospital mortality, and vital signs. Harutyunyan et al. (2019) set variety of benchmark tasks based on the subset of this database. Among them, we conduct binary in-hospital mortality prediction task for measuring classification performance. After preprocessing, our dataset contains 16 features and 21,107 data points.

UCI Localization Data for Person Activity (UCI Human Activity) This dataset includes records of five people doing some usual activities such as walking or sitting. All people wear sensors on their right ankle, left, belt and chest. During activities, the sensors record their position in the form of three-dimensional coordinates at very short intervals. Activities of each people at a certain time point are classified into one of 11 classes and recorded with the position of sensors. After preprocessing, we have total 6554 time series with 12 features (3-dimensional coordinates of 4 devices). Using this data, we conduct an online-prediction task. The objective of this task is to classify each individual's activity per time point based on the position of sensors.

A.1.2. DATA PREPROCESSING

For all datasets, we basically standardize the numerical covariates so that all features have zero mean and unit variance, respectively. Also, we normalize the time scale to be in $[0, 1]$ scale.

PhysioNet2012 and MIMIC-III Since there is no fixed rule for preprocessing PhysioNet 2012 dataset and MIMIC-III database researchers usually preprocess the raw data on their own. Therefore, it is difficult for practitioners to compare experimental results with other works. So, for comparability, we employ python package `medical_ts_datasets`[1] (Horn et al., 2020) which provides the unified data preprocessing pipeline for Physionet2012 and MIMIC-III datasets. For both datasets, patients who have more than 1000 time steps or have no observed time series data were excluded. Also, discretizing the time step of data by an hour and aggregating the measurement is frequently used to preprocess Physionet2012 in previous work (Rubanova et al., 2019), but this package preserves much more original time series variables while preprocessing than hourly-based aggregation preprocessing for both datasets. Please refer to (Horn et al., 2020) for more details of dataset preprocessing.

---

[1]https://github.com/ExpectationMax/medical_ts_datasets

UCI Human Activity   Basically, we decide to follow the preprocessing of this dataset based on (Rubanova et al., 2019), except for aggregation of time points of all data points in the dataset. We just use overlapping intervals of 50 time points while (Rubanova et al., 2019) used 221 time points. So, our processed data finally contains 6554 data points and 50 time points, and 12 features.

A.2. Details for classification experiments

For all experiments, we use five different random seeds to conduct experiments. Then we measure the mean and the variance of each value.

A.2.1. BASELINE METHODS

- **GRU-mean**: Missing value is simply replaced with the empirical mean of each variable.

- **GRU-forward**: Missing entries are filled with previously observed values.

- **GRU-simple**: concatenate the mask s_t, and the *time-interval* δ_t along with the imputed vector $\hat{x}_t$. The concatenated vector $[\hat{x}_t, s_t, \delta_t]$ is then fed into GRU.

- **GRU-D**: Missing values are imputed as a weighted mean of the last observed $x_{t',j}$ and the mean $\bar{x}_j$ with the learnable weight.

- **Phased-LSTM**: This model is LSTM variant designed to deal with long sequence input by introducing a time gate in their cell to prevent memory decay when useful information is absent for a long time (Neil et al., 2016).

- **Interpolation-Prediction Network (IP-Nets)**: Instead of directly imputing missing values, IP-Nets (Shukla & Marlin, 2019) employed semi-parametric interpolation network that makes regularly spaced representation of irregularly sampled time series data. Then, this representation is fed into a prediction network such as GRU.

- **Set Function for Time Series (SeFT)**: SeFT (Horn et al., 2020) is a set function-based method attached with an attention mechanism for multivariate time series classification task. This model encodes the observed value of the time series as a set and uses the attention layer to aggregate the embedding of elements. Since SeFT only uses the observed value as input, imputation is unnecessary.

A.2.2. MODEL IMPLEMENTATION IN HUMAN ACTIVITY EXPERIMENT

Since we conduct online-prediction task on Human Activity dataset, we do not consider IP-Nets as baseline models because this model exploits future information when conducting interpolation before feeds the representation to the classification model. SeFT also use future information due to the set encoding, but the author offers the method to prevent information leakage by calculating attention weights in cumulative manner. Since our method uses causal masked attention layer in encoder and decoder, we do not use future information during generation. For classifier, we use causal mask for attention layer in classifier for this experiment.

A.2.3. TRAINING DETAILS

For all the classification experiments, we fix a batch size of 128. We adopt Adam with weight decay as the optimizer and find the best weight decay for each model using grid search.

We conduct a grid search to find the best hyperparameters of each model. See the table for concrete search space and the best hyperparameters for each model.

We employ early stopping for all classification experiments. We set early stopping patience to 20 epochs and set the valid Area Under ROC curve(AUROC) as the early stopping criterion. Since the label imbalance of PhysioNet 2012 and MIMIC-III is extreme, we over-sample the mortality class to train models on the balanced batches.

A.2.4. HYPERPARAMETERS

We search all hyperparameters in the grid to find the best hyperparameters for each model. For all models, we search the weight decay of the AdamW optimizer in $\{0, 0.01, 0.1, 1\}$ and the number of units (n_units) of model layers in $\{128, 256\}$

if the model accepts that parameter. For GRU variant models, we search the dropout probability of forward pass (dropout) in $\{0, 0.1, 0.2, 0.3\}$, and the dropout probability of recurrent pass (recurrent_dropout) in $\{0, 0.1, 0.2, 0.3\}$. For the SeFT-Attn model, we followed the best hyperparameter settings presented in Horn et al. (2020). See Table 6 for hyperparameter settings of our model and baseline methods for all classification experiments.

Table 6. Hyperparameter settings for classification experiments

Dataset	Method	Hyperparameters
PhysioNet2012	GRU-Mean	weight_decay: 0.1, n_units: 128, dropout: 0.3, recurrent_dropout: 0.1
	GRU-Simple	weight_decay: 1.0, n_units: 128, dropout: 0.0, recurrent_dropout: 0.1
	GRU-Forward	weight_decay: 0.0, n_units: 256, dropout: 0.0, recurrent_dropout: 0.2
	GRU-D	weight_decay: 1.0, n_units: 128, dropout: 0.0, recurrent_dropout: 0.2
	PhasedLSTM	weight_decay: 0.0, n_units: 256, use_peepholes: False, leak: 0.01, period_init_max: 1000.0
	IP-Nets	weight_decay: 1.0, n_units: 128, imputation_stepsize: 1 , reconst_fraction: 0.5
	Seft	weight_decay: 0.0, n_phi_layers: 4, phi_width: 128, phi_dropout: 0.2, n_psi_layers: 2
		psi_width: 64, psi_latent_width: 128, dot_prod_dim: 128, n_heads: 4
		attn_dropout: 0.5, latent_width: 32, n_rho_layers: 2, rho_width: 512
		rho_dropout: 0.0, max_timescale: 100., n_positional_dims: 4
	Ours	weight_decay: 0.0, n_train_latents: 10, n_train_samples: 1, n_test_latents: 20, n_test_samples: 30,
		n_hidden: 128, z_dim: 32, n_units: 128, observe_dropout: 0.4
MIMIC-III	GRU-Mean	weight_decay: 1.0, n_units: 128, dropout: 0.1, recurrent_dropout: 0.2
	GRU-Simple	weight_decay: 1.0, n_units: 256, dropout: 0.0, recurrent_dropout: 0.1
	GRU-Forward	weight_decay: 0.0, n_units: 256, dropout: 0.0, recurrent_dropout: 0.1
	GRU-D	weight_decay: 1.0, n_units: 128, dropout: 0.0, recurrent_dropout: 0.2
	PhasedLSTM	weight_decay: 1.0, n_units: 128, use_peepholes: False, leak: 0.01, period_init_max: 1000.0
	IP-Nets	weight_decay: 1.0, n_units: 256, imputation_stepsize: 1, reconst_fraction: 0.5
	Seft	weight_decay: 0.0, n_phi_layers: 3, phi_width: 64, phi_dropout: 0.1, n_psi_layers: 2
		psi_width: 64, psi_latent_width: 128, dot_prod_dim: 128, n_heads: 4
		attn_dropout: 0.1, latent_width: 256, n_rho_layers: 2, rho_width: 512
		rho_dropout: 0.1, max_timescale: 1000., n_positional_dims: 8
	Ours	weight_decay: 0.0, n_train_latents: 10, n_train_samples: 1, n_test_latents: 20, n_test_samples: 30,
		n_hidden: 128, z_dim: 32, n_units: 128, observe_dropout: 0.3
Human Activity	GRU-Mean	weight_decay: 0.0, n_units: 256, dropout: 0.0, recurrent_dropout: 0.0
	GRU-Simple	weight_decay: 0.0, n_units: 256, dropout: 0.0, recurrent_dropout: 0.0
	GRU-Forward	weight_decay: 0.0, n_units: 256, dropout: 0.0, recurrent_dropout: 0.0
	GRU-D	weight_decay: 0.0, n_units: 256, dropout: 0.0, recurrent_dropout: 0.0
	PhasedLSTM	weight_decay: 0.0, n_units: 256, use_peepholes: False, leak: 0.01, period_init_max: 1000.0
	Seft	weight_decay: 0.0, n_phi_layers: 4, phi_width: 128, phi_dropout: 0.2, n_psi_layers: 2
		psi_width: 64, psi_latent_width: 128, dot_prod_dim: 128, n_heads: 4
		attn_dropout: 0.5, latent_width: 32, n_rho_layers: 2, rho_width: 512
		rho_dropout: 0.0, max_timescale: 100., n_positional_dims: 4
	Ours	weight_decay: 0.0, n_train_latents: 10, n_train_samples: 1, n_test_latents: 10, n_test_samples: 10,
		n_hidden: 128, z_dim: 10, n_units: 128, observe_dropout: 0.2

A.2.5. EVALUATION METRICS

For the classification task, we evaluate all models in terms of both predictive accuracy and predictive uncertainty. We use the area under receiver operating characteristic (AUROC), and the accuracy (ACC) to evaluate the predictive performance. To measure the uncertainty calibration of the model, we use cross entropy (CE), expected calibration error (ECE), and brier score (BS) for comparing calibration.

Accuracy Metrics Accuracy metrics are defined using the following terms, where tp, tn, fn, and fp denote true positive, true negative, false negative, and false positive respectively.

$$\text{accuracy} = \frac{tp + tn}{tp + fp + fn + tn} \tag{28}$$

$$\text{precision} = \frac{tp}{tp + fp} \tag{29}$$

$$\text{recall} = \frac{tp}{tp + fn} \tag{30}$$

$$\text{sensitivity} = \frac{tp}{tp + fn} \tag{31}$$

- **AUROC**: area under receiver operating characteristic, area under sensitivity curve.

A.3. Details for imputation experiments

Basically, We randomly delete 10% of observed data for testing the imputation performance of models. We trained our model with five different seeds and also measured the performance of five different seeds.

A.3.1. BASELINE METHODS

- **Mean**: Replace missing values with global mean.

- **Forward**: Impute missing value with previously observed value

- **GP-VAE**: This model is VAE-based probabilistic imputation method proposed by Fortuin et al. (2020). This method employs GP-prior to encode the temporal correlation in the latent space.

- **SAITS**: This model is self-attention based imputation model which Du et al. (2022) proposed. This employs a combination of multiple layers of transformer encoder to impute multivariate time series data.

- **Ours w/o supervision**: To analyze the effect of supervised signal to the imputation performance, we remove the supervised term from our training objective and train only the generative part of our architecture.

- **Ours w/o dropout**: Our model without obsdropout.

- **Ours w/o supervision and MNAR**: Our model without obsdropout and MNAR assumption, which missing model and classifier is removed from our model.

A.3.2. EVALUATION METRICS

For the imputation task, we evaluate all methods in terms of MRE(Mean Relative Error) and MAE(Mean Absolute Error).

A.3.3. TRAINING DETAILS

We applied early stopping with patience 20 and stopping criterion as validation loss of each model. We randomly erase 10% of the observed data for all datasets and measured the evaluation metrics. See Table 7 for hyperparameter settings of our model and baseline methods for all imputation experiments.

Table 7. Hyperparameter settings for imputation experiments

Dataset	Method	Hyperparameters
PhysioNet2012	Mean	-
	Forward	-
	Ours w/ obsdropout	n_train_latents: 10, n_train_samples: 1, n_test_latents: 10, n_test_samples: 1, n_hidden: 128, z_dim: 32, n_units: 128, observe_dropout: 0.4
	Ours w/o supervision	n_train_latents: 10, n_train_samples: 1, n_test_latents: 10, n_test_samples: 1, n_hidden: 128, z_dim: 32, n_units: 128,
	Ours w/o MNAR	n_train_latents: 10, n_train_samples: 1, n_test_latents: 10, n_test_samples: 1, n_hidden: 128, z_dim: 32, n_units: 128, observe_dropout: 0.4
	GP-VAE	n_hidden: 128, z_dim: 35, length_scale: 7, kernel_scale: 1, n_latents: 1,beta: 1
	SAITS	n_head:4 , num_layer:2 , d_model:256 , MIT:True, input_with_mask: True, MIT_missing_rate:0.4 , n_hidden:128
MIMIC-III	Mean	-
	Forward	-
	Ours w/ obsdropout	n_train_latents: 10, n_train_samples: 1, n_test_latents: 10, n_test_samples: 1, n_hidden: 128, z_dim: 32, n_units: 128, observe_dropout: 0.3
	Ours w/o supervision	n_train_latents: 10, n_train_samples: 1, n_test_latents: 10, n_test_samples: 1, n_hidden: 128, z_dim: 32, n_units: 128,
	Ours w/o MNAR	n_train_latents: 10, n_train_samples: 1, n_test_latents: 10, n_test_samples: 1, n_hidden: 128, z_dim: 32, n_units: 128, observe_dropout: 0.3
	GP-VAE	n_hidden: 128, z_dim: 35, length_scale: 7, kernel_scale: 1, n_latents: 1,beta: 1
	SAITS	n_head:2 , num_layer:2 , d_model:128 , MIT:True, input_with_mask: True, MIT_missing_rate:0.2 , n_hidden:128
Human Activity	Mean	-
	Forward	-
	Ours w/ obsdropout	n_train_latents: 10, n_train_samples: 1, n_test_latents: 10, n_test_samples: 10, n_hidden: 128, z_dim: 32, n_units: 128, observe_dropout: 0.2
	Ours w/o supervision	n_train_latents: 10, n_train_samples: 1, n_test_latents: 10, n_test_samples: 10, n_hidden: 128, z_dim: 32, n_units: 128,
	Ours w/o MNAR	n_train_latents: 10, n_train_samples: 1, n_test_latents: 10, n_test_samples: 10, n_hidden: 128, z_dim: 32, n_units: 128, observe_dropout: 0.2
	GP-VAE	n_hidden: 128, z_dim: 10, length_scale: 7, kernel_scale: 1, n_latents: 1,beta: 1
	SAITS	n_head:2 , num_layer:2 , d_model:256 , MIT:True, input_with_mask: True, MIT_missing_rate:0.4 , n_hidden:128