Tripletformer for Probabilistic Interpolation of Irregularly sampled Time Series

Vijaya Krishna Yalavarthi
ISMLL, University of Hildesheim
Germany
yalavarthi@ismll.uni-hildesheim.de

Johannes Burchert
ISMLL, University of Hildesheim
Germany
burchert@ismll.uni-hildesheim.de

Lars Schmidt-Thieme
ISMLL, University of Hildesheim
Germany
schimdt-thieme@ismll.uni-hildesheim.de

***Abstract*—Irregularly sampled time series data with missing values is a observed in many fields like healthcare, astronomy, and climate science. Interpolation of these types of time series is crucial for tasks such as root cause analysis and medical diagnosis, as well as for smoothing out irregular or noisy data. To address this challenge, we present a novel encoder-decoder architecture called "Tripletformer" for probabilistic interpolation of irregularly sampled time series with missing values. This attention-based model operates on sets of observations, where each element is composed of a triple of time, channel, and value. The encoder and decoder of the Tripletformer are designed with attention layers and fully connected layers, enabling the model to effectively process the presented set elements. We evaluate the Tripletformer against a range of baselines on multiple real-world and synthetic datasets and show that it produces more accurate and certain interpolations. Results indicate an improvement in negative loglikelihood error by up to 32% on real-world datasets and 85% on synthetic datasets when using the Tripletformer compared to the next best model.**

***Index Terms*—Multivariate Time Series, Irregularly Sampled Time Series with Missing Values, Probabilistic Interpolation**

I. Introduction

In domains such as medical applications [1], multivariate time series (MTS) are frequently observed with irregularity. This implies that the variables (or sensors) within the time series are observed at irregular time intervals and often contain missing values during the alignment process. We refer to such time series as Irregularly Sampled Time Series with Missing Values (IMTS). Practitioners aim to identify data that was not collected in the initial phase of data collection for better analysis. As an example, blood glucose levels are typically measured multiple times a day using a glucose meter, but due to a patient's forgetfulness or other reasons, there may be missing data points. The healthcare provider may want to determine the missing glucose level values in order to better understand the patient's glucose control and make more informed treatment decisions. Probabilistic interpolation models can be used to estimate the probable values of the missing glucose level measurements, along with the uncertainty surrounding the predictions allowing the healthcare provider to avoid overly confident predictions. Furthermore, considering that sensor observations inherently involve measurement errors, it becomes essential to quantify the uncertainty associated with interpolated predictions. This is particularly important in diverse fields such as engineering and environmental applications [2].

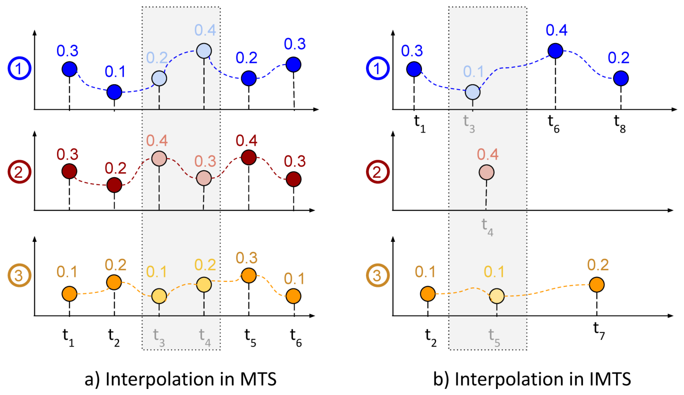


Fig. 1: Interpolation in Multivariate Time Series (a) and Irregularly sampled Time Series (b). In (a) all the channels in the time series are observed at times t_1, t_2, t_5, t_6 and we need to interpolate the values for all the channels at t_3, t_4. Where as in (b), channel 1 is observed at times t_1, t_6, t_8, and channel 3 is observed at t_2, t_7. We did not make any observation in channel 2. However, we need to interpolate the values for channels 1,2 and 3 at time points t_3, t_4, and t_5 respectively.

Deep learning models have been widely studied for imputation tasks in general multivariate time series (MTS) with missing values, where the time series is observed at regular intervals, the sparsity (number of missing values over the number of observed values) is small, and all the channels are of equal lengths. However, these models are not well suited for irregularly sampled time series with missing values (IMTS) due to their unique properties. IMTS often consist of non-periodic observations in a channel, extremely sparse channels when trying to align the IMTS, and variable channel lengths in a single example. These properties make modeling IMTS difficult for standard deep learning models. Figure 1 illustrates the differences between interpolation in IMTS and MTS.

In the domain of IMTS, most research has focused on classification techniques [3], [4], [5], [6], [7], [8], [9]. However, some studies have delved into the area of deterministic interpolation [3], [10], [5], but a few on the probabilistic interpolation. Gaussian Process Regression models [11], [12] were used traditionally for the task [13]. Tashiro et. al.[14] proposed a score based diffusion model, CSDI, however it only produced homoscedastic variance meaning it provides same uncertainty for all the outputs. Recently, Sukla et. al. [15] introduced the Heteroscedastic Variational Autoencoder (HETVAE) which uses the Uncertainty aware multi Time Attention Network (UnTAN) to produce heteroscedastic variance for probabilistic interpolation in IMTS. HETVAE struggles to effectively learn cross-channel interactions because its core component UnTAN is built upon existing multi-Time Attention Network (mTAN) [3] which employs a separate time attention model for each channel. Through our experiments, we have found that encoding observations from all channels using a single encoder leads to more accurate interpolations.

In this work, we propose a novel encoder-decoder architecture, that we call Tripletformer for the probabilistic interpolation of IMTS . As the name suggests, our Tripletformer operates on the observations that are in triplet form (time, channel and value of the observation). We model time and channel as the independent variables, and the observation measurement (value) as the dependent variable. The encoder of the Tripletformer learns the interactions among the inter and intra channel observations of the IMTS simultaneously, while the decoder produces the probability distributions of the dependent variable over the set of reference (or target) independent variables. We overcome the computational complexity bottleneck of canonical attention by using induced multi-head attention block [16] that has near linear computational complexity (that we explain in Section III-E).

We evaluate the proposed Tripletformer over multiple 3 real-world and 2 synthetic datasets at varying observation levels, and two different missing patterns: random missing and burst missing. The Tripletformer is compared with the state-of-the-art probabilistic interpolation models, HETVAE and CSDI, and a range of baselines using Negative Loglikelihood loss (NLL) and CRPS as the evaluation metric. Our experimental results attest that the proposed Tripletformer provides significantly better interpolations compared to its competitors. Our contributions are summarized as follows:

- We propose a novel model called Tripletformer that can learn both inter and intra channel interactions simultaneously in the IMTS by operating on the observations for probabilistic interpolation.
- To overcome the computational complexity of multi-head attention, we employ an induced multi-head attention mechanism [16]. This improves predictions while efficiently addressing the computational bottleneck.
- We perform extensive experimental evaluation on 3 real and 2 synthetic IMTS datasets, under two different missing patterns: random and burst missing. Our results attest that Tripletformer provides up to 32% and 85% improvement in NLL over real and synthetic IMTS datasets, respectively compared to the next best model HETVAE.

We provide the implementation of Tripletformer in https://github.com/yalavarthivk/tripletformer.

II. Related Work

The focus of this work is on probabilistic interpolation of irregularly sampled time series with missing values (IMTS). Majority of the recent works in this field have mainly focused on deterministic interpolation techniques for IMTS.

For example, [17] and [18] applied Multidirectional Recurrent Neural Network and Bidirectional Recurrent Neural Networks for IMTS imputation by using a GRU decay (GRU-D) [19] model as the underlying architecture to handle the irregularity in the series. [8] proposed Interpolation-Prediction networks, which consist of a semi-parametric model to share information across multiple channels of an IMTS. This approach is similar to a multivariate Gaussian process [11], but does not use a positive definite co-variance matrix.

These methods provide deterministic interpolation, which means they cannot be used to find the uncertainty around the predictions, as done in probabilistic interpolation. This is a limitation of these models, as probabilistic interpolation provides more information about the confidence of the predictions, which can be useful in many real-world applications. The current work aims to propose a novel probabilistic interpolation method for IMTS.

Researchers have proposed various models for probabilistic interpolation of time series data that utilize Variational Autoencoders (VAEs) and can produce homoscedastic variance, which means the uncertainty in the predictions is constant across all predictions. In 2018, [10] proposed using a Neural Ordinary Differential Equations (ODE) network for modeling time series by using a continuous time function in the hidden state. Later, in 2019, [5] proposed an architecture with ODE-RNN as the encoder which uses Neural ODE to model the latent state dynamics and an RNN to update it when a new observation is made. In 2020, [20] proposed using stochastic differential equations (SDEs) which generalize ODEs and are defined using non-standard integrals, usually relying on Itô calculus. In 2021, [21] proposed Neural ODE processes (NDPs) by combining Neural ODEs with Neural Processes [22]. NDPs are a class of stochastic processes that are determined by a distribution over Neural ODEs. Other than ODE based models, [3] proposed Multi-Time Attention Networks (mTAN), which uses a temporal attention network at both encoder and decoder to produce deterministic interpolations. Recently in 2021, [14] a score based diffusion model called CSDI was proposed. The main idea behind this model is to use a score function to evaluate the likelihood of different imputations and to select the one that is most accurate.

While VAE and diffusion-based models can produce homoscedastic variance, it is also possible to produce heteroscedastic variance, where the uncertainty in the predictions varies across different predictions. One approach to producing heteroscedastic variance is to output the distribution parameters, rather than a fixed point estimate. Gaussian Process Regression (GPR) [12] provides full joint posterior

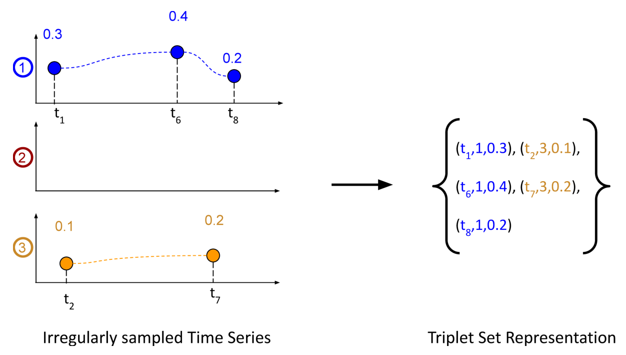


Fig. 2: Demonstration of IMTS as set of observations.

distributions over interpolation outputs, which can be used to produce heteroscedastic variance. GPR models can be directly implemented on IMTS by applying one model for each channel. Multi-task Gaussian Process [11] have been studied for the multivariate setting but they require a positive definite covariance matrix. In [13], Dürichen et.al., studied various kernels for the Multi-task Gaussian Process for the analysis of IMTS. Finally, HETVAE model, proposed in [15] is the first work that deals with the problem of probabilistic interpolation in IMTS and provides heteroscedastic variance. The HETVAE model consists of an Uncertainty-aware multi-Time Attention Networks (UnTAN) that uses a combination of probabilistic and deterministic paths to output heteroscedastic variance.

In this work, an attention-based model called Tripletformer is proposed for probabilistic interpolation of irregularly sampled time series with missing values (IMTS). Tripletformer operates on the set of observations and produces outputs with heteroscedastic variance.

III. Background and Preliminaries

A. IMTS as set of observations

Our set representation of IMTS follows [7]. Given an IMTS dataset $\mathcal{X} = \{X^n\}_{n=1}^N$ of N many examples, an IMTS X^n can be formulated as set of observations x_i^n such that $X^n = \{x_1^n, x_2^n, ...x_{|X^n|}^n\}$. We omit superscript n when the context is clear. Each observation x_i is a triple consisting of time, channel and value (observation measurement), $x_i = (t_i, c_i, u_i)$, with $t_i \in \mathcal{T} = \{\tau_1, \tau_2, ..., \tau_{|\mathcal{T}|}\}$, $\mathcal{T}$ being the set of all the observation times in the series X, $c_i \in \mathcal{C} = \{1, 2, 3, ..., C\}$, $\mathcal{C}$ being the set of channels in all the IMTS dataset $\mathcal{X}$. In Figure 2, we demonstrate the representation of IMTS as set of observations.

B. Probabilistic Interpolation of IMTS

In deterministic interpolation, one wants to estimate the most probable value of the dependent variable. While deterministic interpolation provides a single estimate, probabilistic interpolation outputs the probability distribution that explains the prediction's uncertainty.

Given an Irregularly sampled time series $X \in \mathcal{X}$, independent variables ($w = (t', c')$) in the target observations that are time $t' \notin \mathcal{T}$, $\min(\mathcal{T}) < t' < \max(\mathcal{T})$, channel indicator $c' \in \mathcal{C}$; the goal is to predict the probability distribution of the dependent variable u' i.e., find $\hat{Pr}(u'|X, w)$.

Problem 1 (Probabilistic Interpolation in Time Series)**.** *Given i) a dataset $\mathcal{D}^{train} \sim \rho$ drawn from an unknown distribution ρ, an instance $(\{x\}^*, \{x'\}^*) \in \mathcal{D}^{train}$ with $x, x' \in \Omega$, $x = (t, c, u), x' = (t', c', u')$, $\Omega = \mathbb{R}_+ \times \mathbb{N} \times \mathbb{R}$; $t, t' \in [\mathbb{R}_+, \mathbb{R}_+]$, $t \neq t'$, ii) a loss function $\mathcal{L} : \mathbb{R} \times \mathcal{F} \to \mathbb{R}$ example negative loglikelihood loss, $\mathcal{F}$ denotes a probability distribution. Find a model $f : \Omega^* \times \mathbb{R}_+^* \times \mathbb{N}^* \to \mathcal{F}$ such that the expected loss is minimized $\mathbb{E}_{(\{x\}^*, \{x'\}^*) \sim \rho} \quad \mathcal{L}(\{u'\}^*, f(\{x\}^*, \{(t', c')\}^*))$.*

C. Multi-head Attention

Multi-head attention (MHA) [23] jointly attends different positions from different representation spaces. MHA with h many heads is defined as

$$\text{MHA}(q, k, v) = \text{Concat}(\text{head}_1, ..., \text{head}_h)\theta^o \tag{1}$$

where $q \in \mathbb{R}^{L_q \times d_q}$, $k \in \mathbb{R}^{L_k \times d_k}$, $v \in \mathbb{R}^{L_k \times d_v}$ are called query, key and value sequences with lengths of L_q, L_k, L_k, and embedding dimension of d_q, d_k, d_v respectively. $\text{head}_i, i \leq h$ is the i^{th} head which is defined as:

$$\text{head}_i = \text{Att}(q\theta_i^q, k\theta_i^k, v\theta_i^v) = Softmax(A)v\theta_i^v \tag{2}$$

$$\text{Att} = \frac{q\theta_i^q \left(k\theta_i^k\right)^T}{\sqrt{d}} \tag{3}$$

where $\theta_i^q \in \mathbb{R}^{d_q \times d}$, $\theta_i^k \in \mathbb{R}^{d_k \times d}$, $\theta_i^v \in \mathbb{R}^{d_v \times d}$ and $\theta^o \in \mathbb{R}^{d \times d_o}$ are the learned parameters, d is the hidden dimension of the projection spaces, and d_o is the dimension of the output sequence after MHA.

Attention matrix (A) which requires multiplication of $L_q \times d$ and $L_k \times d$ sized matrices has a computational complexity of $\mathcal{O}(L_q L_k d)$. When both L_q and L_k are arbitrarily large, computing A will be a bottleneck.

D. Multihead Attention Block (MAB)

MAB [23], [16] consists of two sublayers: i) Multihead Attention (MHA) [23], and ii) a pointwise feed forward layer (MLP) as shown in Figure 3(a). We have a residual connection around both the sublayers.

$$\begin{aligned} \text{MAB}(q, k, v) &= \alpha(H + \text{MLP}(H)) \\ \text{where} \quad H &= \alpha(q + \text{MHA}(q, k, v)) \end{aligned} \tag{4}$$

with q, k, v being query, key and value sequences and, α is a non-linear activation function.

In MHA, the attention matrix (A) which requires the multiplication of $L_q \times d$ and $L_k \times d$ sized matrices has a computational complexity of $\mathcal{O}(L_q L_k d)$. L_q, L_k, d are the query length, key length and embedding dimensions respectively. When both L_q and L_k are large, computing A will be a bottleneck because of its quadratic computational complexity.

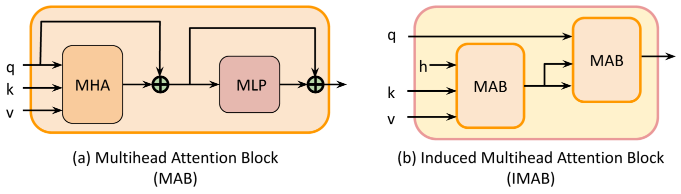

q
k
v
MHA
MLP
(a) Multihead Attention Block (MAB)
q
h
k
v
MAB
MAB
(b) Induced Multihead Attention Block (IMAB)

Fig. 3: Architectures of Multihead Attention Block (a) and Induced Multihead Attention Block (b) [16]

E. Induced Multihead Attention Block (IMAB)

Because computing A in MHA is infeasible for long sequences, [16] proposed the Induced Multihead Attention Block (IMAB) that consists of two MABs and l many d_h dimensional induced points $h \in \mathbb{R}^{l \times d_h}$ which are trainable parameters. In IMAB, the attention happens as follows: first induced points attend to the actual keys and values, later queries attend to the induced points. IMAB is given as:

$$\begin{aligned} \text{IMAB}(q, k, v) &= \text{MAB}(q, \mathcal{H}, \mathcal{H}) \\ \text{where} \quad \mathcal{H} &= \text{MAB}(h, k, v) \end{aligned} \tag{5}$$

The architecture of IMAB is shown in Figure 3(b). The computational complexity of the IMAB is $\mathcal{O}(L_q ld + L_k ld)$ which is near linear when $l \ll L_q, L_k$.

IMAB is the optimal choice of model because i) it reduces the computational complexity and ii) provides learned restricted attention phenomena. Probsparse attention [24] which provides restricted attention is the widely used model for the time series forecasting. It ranks the key tokens by their impact on the query, and restricts its attention to the top-K tokens only rather than the entire key set. It was shown that using restricted attention provides similar or better results compared to the canonical attention (see [24] for more details). However, the main drawback of Probsparse attention is, it assumes the lengths of all the series in a batch are equal which is not the case while working with IMTS. In order to have a consistent batch, we pad the series of smaller lengths with zeros, and the sparse attentions treat observations and zero padding in the same manner for sampling the important key tokens which is incorrect. On the other hand, when we are operating on sets, it is not trivial to find the key tokens that are important to the query, manually. Hence, one can learn a set of clusters called reference clusters ($\mathcal{H}$ in equation 5) using h that act as basis for the restriction. Then, the queries attend to those reference clusters rather than the entire key set.

IV. Proposed Model: Tripletformer

Our proposed Tripletformer holds an encoder-decoder architecture for the problem of probabilistic interpolation in IMTS. The model utilizes the irregular sampling of observations by operating on the observations directly.

Our encoder E, similar to that of the transformer model [23], maps the input time series X which is a set of $s = |X|$ many observations $\{x_1, ..., x_s\}$ to a set of continuous representations $Z^{(e)} = \{z_1^{(e)}, ..., z_s^{(e)}\}$ corresponding to X. Our decoder outputs the parameters of the probability distribution of the target observation value $\hat{Pr}(u'|Z^{(e)}, w)$ conditioning up on the encoder output $Z^{(e)}$, and independent variables $w = (t', c')$ of the target observation x'. The architecture of the proposed model is shown in Figure 4. The model components are explained in more detail in the following sections of the paper.

A. Encoder

Our encoder consists of i) an input feed forward embedding layer (iFF) and ii) a Self-attention layer (SA) which is an IMAB.

The iFF in the encoder is a point-wise feed-forward layer that provides learned embeddings to the set elements. To be able to work with the iFF, each observation x in the input time series X is converted into a vector $\mathrm{x} \in \mathbb{R}^{C+2}$ by concatenating the time, channel indicator, and value of the observation. The channel indicator is one-hot encoded. The iFF takes the set of vectors X as input and outputs its embeddings $Y^{(e)} = y_1^{(e)}, ..., y_s^{(e)}$. These input embeddings are then passed through a Self-Attention layer SA which consists of L Induced Multihead Attention Blocks. This produces the latent embeddings $Z^{(e)} = z_1^{(e)}, ..., z_s^{(e)}$.

Using an attention mechanism is a principled approach for the interpolation problem in IMTS. In IMTS, the length of a series (size of the set) varies, and a model that does not depend on the length of the series is needed in order to learn the latent embeddings. One could sort the set elements over time and apply a Convolutional Neural Network (CNN) or Recurrent Neural Network (RNN), but these methods are not useful for interpolation tasks because the query time point is in between the observation times where convolution (or RNN) is already applied. The attention mechanism, on the other hand, allows each observation to attend to all other observations in the set, making it an ideal choice for interpolation tasks.

B. Decoder

The decoder takes the encoder output $Z^{(e)}$, and r many target queries $W = (w_1, ..., w_r)$ as inputs. It then outputs the distribution parameters of the target values $U' = u'_1, ..., u'_r$. Specifically, the decoder outputs $\mathrm{M} = \mu_1, ..., \mu_r$, and $\Sigma = \sigma_1, ..., \sigma_r$ which are the corresponding mean and standard deviation assuming the underlying distribution is Gaussian. In the current work we assume that the underlying distribution is Gaussian but one could output any parametric distribution by simply changing the loss function and the output nodes.

Our decoder consists of three main components: i) a target feed forward embedding layer (tFF), ii) a Cross-attention block (CA) and iii) an output layer (O). The tFF layer maps the target queries to a set of continuous representations, the CA block applies attention mechanism to align the target representations with encoder output, and the O layer produces the mean and standard deviation of the target values.

The target embedding layer tFF is a point wise feed forward layer that provides latent representation to the learned embedding of the target query. For a target query $w \in W$,

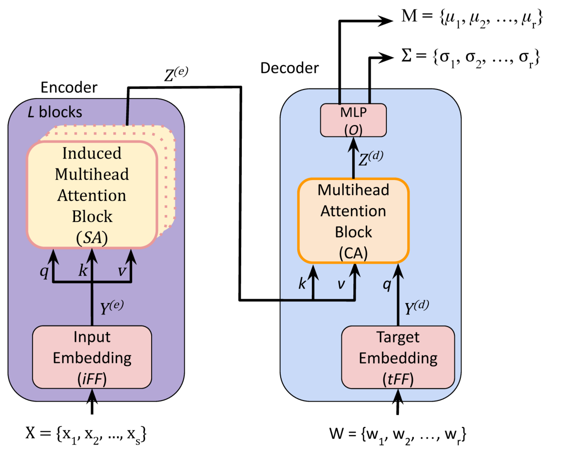


Fig. 4: Tripletformer architecture. Encoder (left) takes the set of observations X, and output their embeddings $Z^{(e)}$. Decoder (right) takes $Z^{(e)}$, target queries (W), and produces the mean M and standard deviation Σ corresponding to W.

we concatenate the time t' and channel indicator c' (one hot encoding), and get $\mathrm{w} \in \mathbb{R}^{C+1}$. $\mathrm{W} = \{\mathrm{w}_1, ..., \mathrm{w}_\mathrm{r}\}$ is passed through tFF to obtain their latent representation $Y^{(d)} = \{y_1^{(d)}, ..., y_r^{(d)}\}$. The Cross-attention layer (CA) takes the $Y^{(d)}$ as query, and $Z^{(e)}$ as keys and values for a MAB, and outputs the learned embeddings $Z^{(d)} = \{z_1^{(d)}, ..., z_r^{(d)}\}$. Finally, $Z^{(d)}$ is passed through the output layer O which is a feed forward layer with two output heads. The output layer produces the distribution parameters M and Σ corresponding to the elements present in W. We assume, for a predictor $w \in W$, $\hat{Pr}(u'|X, w)$ defines the final probability distribution of u' (target observation value) with $\hat{Pr}(u'|X, w) = \mathcal{N}(u'; \mu, \sigma)$.

The following equations describe the flow of data in the model. E_Λ indicates embedding dimension of layer Λ with $\Lambda \in \{iFF, SA, tFF, CA\}$.

$$iFF : \mathrm{X} \mapsto Y^{(e)} \quad \left(\in \mathbb{R}^{s \times E_{iFF}}\right) \tag{6}$$
$$SA : Y^{(e)} \mapsto Z^{(e)} \quad \left(\in \mathbb{R}^{s \times E_{SA}}\right) \tag{7}$$
$$tFF : \mathrm{W} \mapsto Y^{(d)} \quad \left(\in \mathbb{R}^{r \times E_{tFF}}\right) \tag{8}$$
$$CA : Y^{(d)}, Z^{(e)} \mapsto Z^{(d)} \quad \left(\in \mathbb{R}^{r \times E_{CA}}\right) \tag{9}$$
$$O : Z^{(d)} \mapsto \mathrm{M}, \Sigma \quad (\in \mathbb{R}^r, {\mathbb{R}_+}^r) \tag{10}$$
$$u' \sim \hat{Pr}(u'|X, w) = \mathcal{N}(u'; \mu, \sigma), \tag{11}$$
$$\text{where} \quad w = (t', c') \in W, \quad \mu \in \mathrm{M}, \quad \sigma \in \Sigma$$

In certain transformer models [23], [24] for time series forecasting, the decoder employs a self-attention layer on target query embeddings. This introduces a transductive bias among independent variables (covariates). In our study, we lack covariates apart from observation time and channel. As a result, self-attention among target queries is unnecessary. Additionally, applying positional embeddings [23] to both encoder and decoder inputs is not needed. Our triplet already includes time and channel data, which naturally serves as positional embeddings for observations.

In the proposed Tripletformer, we use IMAB in encoder and MAB in the decoder. Although IMAB can be used for both Self-attention (in the encoder) and Cross-attention (in the decoder), we observed that using IMAB for Cross-attention is not providing consistent advantage over MAB in terms of prediction accuracy (see Section V-I2).

C. Supervised learning

Our Tripletformer works in a heteroscedastic manner outputting probability distribution for each query. We assume that the data is following a Gaussian distribution, and use NLL as the main loss for training the model. In addition to NLL, we also use mean square error as the augmented loss in order to avoid the model sticking in local optima when the mean is almost flat after a few iterations. We optimize the following loss function $\mathcal{L}$:

$$\mathcal{L} = \sum_{n=1}^{N} -\mathbb{E}[\log \hat{Pr}(U'^n|X^n, W^n)] + \lambda \mathbb{E}||U'^n - \mathrm{M}^n||_2^2$$
$$\text{with} \quad \hat{Pr}(U'^n|X^n, W^n) = \prod_{j=1}^{r^n} \hat{Pr}(u'^n_j|X^n, w^n_j) \tag{12}$$

V. Experiments

TABLE I: Descrtiption of datasets used in our experiments. Sparsity indicates the % of missing observations when the time series is aligned.

Dataset	#Sample	#Channel	Avg. sparsity	Max.#obs.	Min.#obs.
Physio'12	8,000	41	86	1154	29
MIMIC-III	21,000	17	67	1143	7
Physio'19	40,100	38	81	3080	13
PenDigits	11,000	11	80	8	8
Phon.Spec.	6700	5	91	217	217

A. Datasets

We use the following real world IMTS datasets: *Physionet2012 [25], [26]* Data from 8,000 ICU patients, covering up to 41 measurements taken within the first 48 hours after admission. *MIMIC-III [27]* ICU records from around 21,000 stays, featuring irregularly sampled measurements and 17 observed variables. Dataset split using procedures from [7], [28]. *Physionet2019 [29]* For sepsis early detection, data from about 40,000 ICU samples across three U.S. hospitals, with 38 observed variables.

We also created two synthetic datasets from real world MTS namely PenDigits and PhonemeSpectra [30] (see Section V-F) in order to verify the performance of the TripletFormer in extremely sparse scenario. We provide basic statistics of the datasets used for experiments in Table I.

B. Competing Models

Our Tripletformer competes with the following models. First, we compare with deterministic models that were made probabilistic by adding a homoscedastic variance which is a hyperparameter searched on the validation dataset.

Mean Imputation always predicts the mean value of the channel in the training dataset. *Forward Imputation* predicts

TABLE II: Results on real IMTS datasets, **observations missing at random**. Evaluation measure is NLL, lower the best.

	Obeserved %	10%	50%	90%
Physionet 2012	Mean	1.406	1.406	1.402
	Forward	1.371	1.217	1.164
	GPR	0.956±0.001	0.767±0.006	0.798±0.004
	M-GPR	1.249±0.005	1.191±0.020	1.197±0.055
	L-ODE-RNN	1.268±0.012	1.233±0.027	1.280±0.029
	L-ODE-ODE	1.211±0.001	1.168±0.001	1.170±0.003
	mTAN	1.110±0.000	0.934±0.001	0.923±0.002
	HETVAE	0.849±0.008	0.578±0.005	0.551±0.011
	Tripletformer	**0.780±0.013**	**0.455±0.011**	**0.373±0.040**
	% Improvement	8.1%	21.2%	32.3%
MIMIC-III	Mean	1.508	1.508	1.507
	Forward	1.750	1.423	1.328
	GPR	1.201±0.003	0.979±0.006	0.919±0.001
	M-GPR	1.695±0.041	1.302±0.068	1.297±0.121
	mTAN	1.209±0.000	1.066±0.001	1.065±0.001
	HETVAE	1.077±0.003	0.828±0.001	0.774±0.008
	Tripletformer	**1.056±0.006**	**0.789±0.006**	**0.710±0.010**
	% Improvement	1.9%	4.7%	8.3%
Physionet 2019	Mean	1.421	1.420	1.425
	Forward	1.361	1.205	1.433
	GPR	1.136±0.000	0.907±0.012	0.851±0.007
	mTAN	1.152±0.001	0.988±0.002	0.982±0.006
	HETVAE	1.091±0.001	0.855±0.002	0.835±0.005
	Tripletformer	**1.079±0.001**	**0.806±0.004**	**0.752±0.007**
	% Improvement	1.1%	5.7%	9.9%

TABLE III: Results of **bursts of observations missing** on real IMTS datasets. Evaluation measure is NLL, lower the best.

	Obeserved %	10%	50%	90%
Physionet 2012	Mean	1.412	1.382	1.407
	Forward	1.534	1.442	1.232
	GPR	1.127±0.018	0.983±0.002	0.691±0.001
	M-GPR	1.351±0.023	1.255±0.037	1.221±0.030
	L-ODE-RNN	1.270±0.002	1.382±0.006	1.289±0.011
	L-ODE-ODE	1.263±0.000	1.242±0.002	1.243±0.001
	mTAN	1.200±0.000	1.141±0.001	0.960±0.001
	HETVAE	1.028±0.011	0.882±0.002	0.614±0.039
	Tripletformer	**0.925±0.005**	**0.777±0.013**	**0.578±0.034**
	% Improvement	11.1%	11.9%	5.8%
MIMIC-III	Mean	1.509	1.507	1.515
	Forward	1.870	1.608	1.412
	GPR	1.328±0.000	1.231±0.004	1.000±0.039
	M-GPR	1.637±0.08	1.42±0.131	1.535±0.138
	mTAN	1.319±0.001	1.247±0.000	1.094±0.002
	HETVAE	1.210±0.000	1.126±0.003	0.939±0.009
	Tripletformer	**1.193±0.003**	**1.087±0.006**	**0.885±0.032**
	% Improvement	1.4%	3.6%	5.8%
Physionet 2019	Mean	1.425	1.415	1.419
	Forward	1.466	1.368	1.209
	GPR	1.257±0.001	1.049±0.002	0.975±0.005
	mTAN	1.269±0.001	1.113±0.001	1.042±0.002
	HETVAE	1.241±0.001	1.012±0.004	0.949±0.002
	Tripletformer	**1.222±0.001**	**0.977±0.005**	**0.865±0.009**
	% Improvement	1.5%	3.5%	8.9%

the value of the previous observation in the channel. If there is no value observed in that particular channel, which is usual for IMTS, we impute with zero. *L-ODE-RNN* [10] is a Neural Ordinary Different Equation (ODE) model where the decoder is an ODE-RNN while the encoder is made of an RNN. *L-ODE-ODE* [5] is also an ODE model where the encoder consists of an ODE-RNN instead of an RNN. *mTAN* [3], Multi-Time attention Network that provides state-of-the-art results for deterministic interpolation, uses time attention networks for the series encoding and decoding.

We also compare with the following probabilistic models: *GPR* [12] is the Gaussian Process regression modality where one model is trained per channel, *M-GPR [13]* is a multi-task Gaussian Process model where all the channels are trained simultaneously using a single architecture employing convolution of kernels, *HETVAE [15]*, Heteroscedastic Temporal Variational Autoencoder for Irregular Time Series (HETVAE) is the state-of-the-art probabilistic interpolation model for IMTS. *CSDI [14]*, a continuous score based model for probabilistic imputation of IMTS was evaluated for Continuous Ranked Probability Score (CRPS). Hence, we did a separate experiment to compare TripletFormer with CSDI for CRPS.

C. Experimental Setup

We randomly split each dataset into train and test with 80% and 20% samples. Further 20% of the train set is used as validation set. We search the following hyperparameters: $L \in \{1, 2, 3, 4\}$, hidden units in MLP layers (feed forward layers in both encoder and decoder) from $\{64, 128, 256\}$, attention layers dimension from $\{64, 128, 256\}$, number of induced points in IMAB from $\{16, 32, 64, 128\}$, and λ augmented loss weight from $\{0, 1, 5, 10\}$. Similar to [7], we trained all the competing models for 5 different random hyperparameter setups, and chose the one that has the least NLL on the validation dataset. We independently run the experiments for 5 times using the best hyperparameters and different random seeds. We code all the models using PyTorch, and trained on NVIDIA GeForce RTX 3090 GPUs.

Data preprocessing: We found that some observations in the datasets contain errors. For example, in the Physionet dataset, we noticed that some of the Ph Value readings were around 700 which is outside the normal range of 0 to 14. To address this issue, we implemented a standard practice for handling real-world IMTS datasets by identifying and removing observations that fall outside the 99.9^{th} percentile of the observation range in the training data. To further preprocess the data, we also rescaled the time values between 0 and 1 for all datasets and standardized each channel to the standard normal distribution.

D. Task setting

In the experiments, we considered two types of setups for the interpolation task.

Random missing: In this setup, we assume that the sensors are missed at random time points. Hence we condition on the random time points and predict the observation values of all the available channels for the remaining time points.

Burst missing: Here, we assume that the sensors are not observed for a series of time points. We randomly choose a start time point, and use p many time points after that for prediction by conditioning upon the remaining.

E. Results on real IMTS datasets

Our main experimental results using two different sampling techniques on real IMTS datasets are presented below.

1) Results on random missing: In Table II, we present the results for the random missing setup. Three different conditioning ranges (observed %) which are $10\%, 50\%$ and 90% of all the observations in the series were used. We could not run ODE models on MIMIC-III and Physionet2019 datasets due to high computational requirement. However, for the datasets where they have been evaluated, ODE models performed significantly worse than Tripletformer. Hence, it is safe to assume that ODE models perform worse in the remaining datasets as well. Similarly, we could not compute M-GPR on Physionet2019 due to limited computational resources as it requires computing the inverse of co-variance matrix.

As also observed in [15], we notice that M-GPR performs worse than GPR. We see that our Tripletformer outperforms all the competing models with a huge margin. Especially, compared to the next best model, HETVAE, significant lifts are observed in the Physionet2012 dataset.

We note that the published results [15] with 50% time points in conditioning range using random missing for both the Physionet2012 and MIMIC-III differ from ours. For the Physionet2012, this can be attributed to the following: 1) in [15], the dataset was normalized using the parameters computed on the entire dataset instead of training dataset, 2) inclusion of erroneous observations in [15] (see Section V-C). Also, we were not able to reproduce the results with the preprocessing details provided in [15]. Whereas, for MIMIC-III dataset, we could not extract the splits used in [15], hence used another standard version [7], [28].

We would like to note that the proposed Tripletformer is scalable to the large time series datasets. IMAB consists of near-linear computational complexity as explained in Section III-E making the model scalable. As an example, Tripletformer can seamlessly process Physionet2019 dataset which has up to 3080 observations.

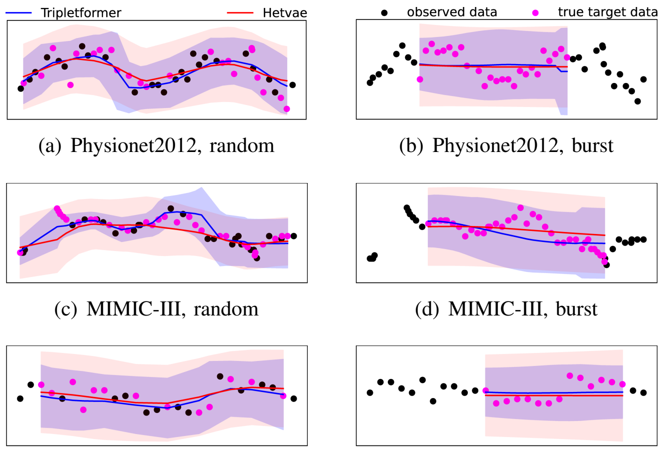
2) Results on burst missing: We present the results for burst sampling in Table III. Interpolating the burst missing values is a difficult task compared to that of random missing. Because, in burst missing, conditioning time points are far from the query time point whereas in random sampling they are close by. The results on burst sampling show a similar pattern to random sampling. Our Tripletformer outperforms all the models with a significant margin; HETVAE and GPR are the second and third best models. We see significant lifts in this setup as well compared to the state-of-the-art model, HETVAE. Again, in Physionet2012 dataset with 50% of the observations in the conditioning range, our Tripletformer has around 12% improvement in NLL.

We perform a qualitative comparison of the predictions made by the Tripletformer and HETVAE models in Figure 5. We observe that the Tripletformer provides not only accurate predictions but also with higher certainty which is desirable in probabilistic interpolation.

F. Experiments with synthetic IMTS datasets

In this experiment, we see the performance of the competing models in the extremely sparse setup. We set MTS to be observed asynchronously meaning each senor is observed independent of others. Hence, we assume that at every point of time only one sensor is observed. From MTS, we randomly chose one variable at a single point of time, and remove all the remaining variables making the number of observations in synthetic IMTS is same as the length of the source MTS.

For this, we choose PenDegits and PhonemeSpectra which are the second and third largest datasets used in [30]. While the largest dataset, FaceDetection, lacked variability in interpolation results across models and sampling techniques, PenDigits and PhonemeSpectra were chosen as suitable alternatives. Experimental results for random and burst sampling are presented in Tables IV and V respectively. For PenDigits, both sampling techniques yielded the same results due to the short time series length when 90% of the series is observed. For both the



(a) Physionet2012, random (b) Physionet2012, burst (c) MIMIC-III, random (d) MIMIC-III, burst (e) Physionet2019, random (f) Physionet2019, burst

Fig. 5: Comparison of qualitative performance between Tripletformer and HETVAE. Plots are the predictions (95^{th} quantile) of both the models for heart rate in all the three datasets conditioned on the 50% of the time points.

TABLE IV: Results on synthetic IMTS, **observations missing at random**. Evaluation measure is NLL, lower the best.

	Obeserved %	10%	50%	90%
Pen Digits	Mean	1.424	1.420	1.424
	Forward	1.550	1.626	1.596
	GPR	1.411±0.001	1.350±0.030	1.279±0.055
	M-GPR	1.400±0.001	1.326±0.059	1.558±0.474
	L-ODE-RNN	1.285±0.005	1.183±0.012	1.102±0.014
	L-ODE-ODE	1.307±0.001	1.182±0.002	1.108±0.004
	mTAN	1.307±0.001	1.050±0.013	0.882±0.020
	HETVAE	1.267±0.003	1.207±0.025	1.326±0.015
	Tripletformer	**1.115±0.003**	**0.693±0.019**	**0.463±0.027**
	% Improvement	11.9%	42.6%	65.1%
Phoneme Spectra	Mean	1.437	1.437	1.435
	Forward	1.553	1.559	1.528
	GPR	1.378±0.002	1.271±0.005	1.176±0.009
	M-GPR	1.301±0.007	1.116±0.077	1.961±1.385
	L-ODE-RNN	1.327±0.006	1.289±0.010	1.263±0.025
	L-ODE-ODE	1.304±0.002	1.273±0.006	1.264±0.008
	mTAN	1.225±0.003	0.963±0.009	1.097±0.010
	HETVAE	1.033±0.031	0.699±0.004	0.804±0.002
	Tripletformer	**0.923±0.008**	**0.413±0.005**	**0.115±0.024**
	% Improvement	10.7%	40.9%	85.7%

TABLE V: Results of synthetic IMTS for **observations missing at burst**. Evaluation measure is NLL, lower the best.

	Obeserved %	10%	50%	90%
Pen Digits	Mean	1.427	1.396	1.424
	Forward	1.556	1.683	1.596
	GPR	1.386±0.006	1.413±0.000	1.279±0.055
	M-GPR	1.381±0.010	1.345±0.012	1.344±0.218
	L-ODE-RNN	1.271±0.002	1.184±0.005	1.144±0.007
	L-ODE-ODE	1.269±0.002	1.207±0.020	1.171±0.075
	mTAN	1.277±0.002	1.061±0.004	0.882±0.023
	HETVAE	1.270±0.001	1.188±0.006	1.326±0.033
	Tripletformer	**1.115**±**0.000**	**0.787**±**0.015**	**0.463**± **0.027**
	% Improvement	12.2%	33.8%	65.1%
Phoneme Spectra	Mean	1.441	1.436	1.456
	Forward	1.583	1.643	1.610
	GPR	1.387±0.000	1.331±0.000	1.309±0.001
	M-GPR	1.469±0.101	1.345±0.035	1.305±0.011
	L-ODE-RNN	1.389±0.010	1.356±0.013	1.309±0.002
	L-ODE-ODE	1.367±0.001	1.343±0.002	1.290±0.001
	mTAN	1.350±0.001	1.285±0.001	1.314±0.008
	HETVAE	1.237±0.003	1.057±0.002	1.006±0.002
	Tripletformer	**1.180**±**0.008**	**1.025**±**0.005**	**0.975**±**0.009**
	% Improvement	4.6%	3.2%	3.1%

TABLE VI: Comparison of Tripletformer and HETVAE with published results from [14] on Physionet2012 dataset and random missing. Evaluation measure is CRPS, lower the best. † indicates published results from [14].

Observed%	10%	50%	90%
L-ODE-ODE	0.761±0.010†	0.676±0.003†	0.700±0.002†
mTAN	0.689±0.015†	0.567±0.003†	0.526±0.004†
HETVAE	0.188±0.007	0.116±0.016	0.256±0.003
CSDI	0.556±0.003†	0.418±0.001†	0.380±0.002†
Tripletformer	**0.062**±**0.006**	**0.062**±**0.009**	**0.061**±**0.010**

sampling types, and all the conditioning ranges, Tripletformer provides superior interpolations.

The reason for the poor performance of the HETVAE on synthetic datasets is the encoding of each channel separately. We note that the medical datasets do not have significant cross channel interactions compared to the synthetic IMTS datasets making HETVAE shine comparatively better in the former. However, *for both synthetic and real IMTS datasets, Tripletformer outperforms HETVAE significantly*.

G. Comparison with CSDI [14]

CSDI is a score based diffusion model that produce probabilistic outputs. Here, we compare the Tripletformer with CSDI in terms of Continuous Ranked Probability Score (CRPS) as shown in [14]. Because of huge computational requirement of diffusion models, conducting experiments with CSDI on all the datasets is beyond our computational resources, hence, we compare both Tripletformer and HETVAE on the published results from [14] in Table VI. For fair comparison, we use the same data splits of Physionet2012 data that was given in [14]. We note that CSDI produces outputs with homoscedastic variance similar to that of mTAN and L-ODE-ODE. It an be observed that the Tripletformer outperforms all the baseline models by significant margin for the CRPS score as well. On an average, *Tripletformer improves the interpolation accuracy by* 63% *and* 86% *respectively compared to HETVAE and CSDI models respectively*.

H. Experiment on Deterministic Interpolation

TABLE VII: Comparison of competing models for deterministic interpolation on Physionet2012 dataset and random missing. Evaluation metric Mean Squared Error, lower the best. † indicates published results from [3].

Model	Mean Squared Error $(\times 10^{-3})$		
L-ODE-RNN	8.132±0.020†	8.171±0.030†	8.402±0.022†
L-ODE-ODE	6.721±0.109†	6.798±0.143†	7.142±0.066†
mTAN	4.139±0.029†	4.157±0.053†	4.798±0.036†
HETVAE	4.200±0.500	4.945±0.000	4.600±0.000
Tripletformer	**3.500**±**0.100**	**3.600**±**0.100**	**3.900**±**0.200**
Observed %	50%	70%	90%

It is interesting to see the performance of Tripletformer for the deterministic interpolation. We trained both Tripletformer and HETVAE for predicting the mean value (optimized for Mean Squared Error loss). Tripletformer is compared with HETVAE [15] (probabilistic model), mTAN [3], L-ODE-ODE [5] and L-ODE-RNN [10] models on Physionet2012 dataset for random missing setup in Table VII. While we did an experiment for HETVAE, published results from [3] were reported for mTAN and ODE models because we use the same dataset splits provided by [3]. Notably *Tripletformer outperforms all the models with a significant margin for deterministic interpolation as well, and reduced the mean squared error of the next best model by* 12% *on an average*.

I. Ablation Study

1) Tripletformer vs. TF-enc-MAB: In table VIII, we compare the performance of Tripletformer with its variant Tf-enc-MAB which consists of MAB in the encoder. We use synthetic IMTS datasets for comparison because we could not run Tf-enc-MAB on all the splits of real IMTS datasets. We see that choosing IMAB instead of MAB in the encoder is optimal. As mentioned in Section III-E, Tripletformer that has IMAB in the encoder provides similar or better performance compared to Tf-enc-MAB demonstrating the advantage of restricted attention in IMAB. We see significant lifts when conditioning range increases from 10% to 90%.

TABLE VIII: Comparing Tripletformer and Tf-enc-MAB. Tf-enc-MAB is a variant of Tripletformer and consists of the MAB in the encoder instead of the IMAB. Results of synthetic IMTS. Evaluation measure is NLL, lower the best.

	tp. obs.	10%	50%	90%
		random sampling		
Pen Digits	Tf-enc-MAB	1.117±0.001	0.713±0.009	0.551±0.026
	Tripletformer	**1.115**±**0.003**	**0.693**±**0.019**	**0.463**±**0.027**
Phoneme Spectra	Tf-enc-MAB	0.966±0.010	0.626±0.142	0.504±0.156
	Tripletformer	**0.923**±**0.008**	**0.413**±**0.005**	**0.115**±**0.024**
		burst sampling		
Pen Digits	Tf-enc-MAB	1.111±0.002	0.838±0.020	0.551±0.026
	Tripletformer	**1.115**±**0.000**	**0.787**±**0.015**	**0.463**± **0.027**
Phoneme Spectra	Tf-enc-MAB	1.205±0.009	1.035±0.004	1.019±0.026
	Tripletformer	**1.180**±**0.008**	**1.025**±**0.005**	**0.975**±**0.009**

2) Tripletformer vs. Tf-dec-IMAB: We compare the Tripletformer with Tf-dec-IMAB which is a variant of Tripletformer where MAB is replaced with IMAB in the decoder. Comparisons are made using all the datasets for both random and burst sampling and the results are presented in Table IX. We see that, among 17 out of 30 comparisons Tripletformer performs better. Among real IMTS datasets, Tf-dec-IMAB provides better performance in burst sampling for MIMIC-III with 50% and 90% time points and Physionet2019 in 90% time points in conditioning range. Whereas Tripletformer has better performance in Physionet2012 dataset for both sampling techniques when conditioned up of 50% and 90% of the available time points. We see similar behavior for synthetic IMTS datasets as well. While Tf-dec-IMAB performs better for PenDigits, Tripletformer performs better in PhonemeSpectra. We see that though Tf-dec-IMAB predicts well in some scenarios, on an average Tripletformer provide around 5% less error.

TABLE IX: Comparing Tripletformer and Tf-dec-MAB (variant of Tripletformer, consists the IMAB in the decoder instead of the MAB). Evaluation measure is NLL, lower the best.

	tp. obs.	10%	50%	90%
		random sampling		
Physionet'12	Tf-dec-IMAB	0.845±0.015	0.565±0.05	0.703±0.092
	Tripletformer	**0.780±0.013**	**0.455±0.011**	**0.373±0.040**
MIMIC-III	Tf-dec-IMAB	**1.053±0.037**	**0.783±0.01**	0.735±0.018
	Tripletformer	1.056±0.006	0.789±0.006	**0.710±0.010**
Physionet'19	Tf-dec-IMAB	**1.061±0.003**	**0.796±0.002**	0.797±0.017
	Tripletformer	1.079±0.001	0.806±0.004	**0.752±0.007**
Pen Digits	Tf-dec-IMAB	1.113±0.001	0.717±0.028	**0.386±0.054**
	Tripletformer	**1.115±0.003**	**0.693±0.019**	0.463±0.027
Phoneme Spectra	Tf-dec-IMAB	0.951±0.009	0.570±0.174	0.249±0.042
	Tripletformer	**0.923±0.008**	**0.413±0.005**	**0.115±0.024**
		burst sampling		
Physionet'12	Tf-dec-IMAB	0.946±0.007	0.821±0.008	0.794±0.023
	Tripletformer	**0.925±0.005**	**0.777±0.013**	**0.578±0.034**
MIMIC-III	Tf-dec-IMAB	1.203±0.003	**1.023±0.006**	**0.835±0.015**
	Tripletformer	**1.193±0.003**	1.087±0.006	0.885±0.032
Physionet'19	Tf-dec-IMAB	**1.210±0.001**	0.980±0.003	0.797±0.017
	Tripletformer	1.222±0.001	**0.977±0.005**	0.865±0.009
Pen Digits	Tf-dec-IMAB	1.127±0.001	**0.735±0.015**	**0.386±0.054**
	Tripletformer	**1.115±0.000**	0.787±0.015	0.463± 0.027
Phoneme Spectra	Tf-dec-IMAB	1.217±0.007	1.032±0.011	**0.950±0.011**
	Tripletformer	**1.180±0.008**	**1.025±0.005**	0.975±0.009

J. Additional Experiments for deterministic interpolation

In Table X, we present the results for deterministic interpolation on all the datasets for mTAN, HETVAE and Tripletformer. We observe that Tripletformer provides best performance in 24 out of 30 comparisons.

K. Discussions

While Tripletformer offers probabilistic interpolation using parametric distributions, it is susceptible to performance degradation if the assumed distribution mismatches the observed data.

Additionally, Tripletformer focuses on providing marginal distributions rather than joint distributions. The focus on joint distributions can lead to variability in results when arbitrary time points are included. To address these limitations, we plan to develop a model that employs joint distributions, potentially enhancing prediction smoothness and reducing variability for random target times.

CONCLUSIONS

In this work, we propose a novel model called Tripletformer for the problem of probabilistic interpolation in Irregularly sampled Time Series with missing values (IMTS). Our Tripleformer consists of a transformer like architecture, operates on a set of observations. We employ induced multi-head attention block in the encoder in order to learn the restricted attention mechanism, and to circumvent the problem of quadratic computational complexity of the canonical attention. We experimented on 5 real and synthetic IMTS datasets, various conditioning ranges and 2 different sampling techniques: random missing and burst missing. Our experimental results attest that the proposed Tripletformer provides better interpolations compared to the state-of-the-art model HETVAE.

ACKNOWLEDGMENTS

This work was supported by the Federal Ministry for Economic Affairs and Climate Action (BMWK), Germany, within the framework of the IIP-Ecosphere project (project number: 01MK20006D), and Funded by the Lower Saxony Ministry of Science and Culture under grant number ZN3492 within the Lower Saxony "Vorab" of the Volkswagen Foundation and supported by the Center for Digital Innovations (ZDIN).

REFERENCES

[1] P. Yadav, M. Steinbach, V. Kumar, and G. Simon, "Mining electronic health records (ehrs) a survey," *ACM Computing Surveys (CSUR)*, vol. 50, no. 6, pp. 1–40, 2018.
[2] M. Lepot, J.-B. Aubin, and F. H. Clemens, "Interpolation in time series: An introductive overview of existing methods, their performance criteria and uncertainty assessment," *Water*, vol. 9, no. 10, p. 796, 2017.
[3] S. N. Shukla and B. Marlin, "Multi-time attention networks for irregularly sampled time series," in *International Conference on Learning Representations*, 2021.
[4] S. C.-X. Li and B. Marlin, "A scalable end-to-end gaussian process adapter for irregularly sampled time series classification," *Advances in Neural Information Processing Systems 29*, 2016.
[5] Y. Rubanova, R. T. Chen, and D. Duvenaud, "Latent odes for irregularly-sampled time series," *Advances in Neural Information Processing Systems 32*, 2019.
[6] P. Kidger, J. Morrill, J. Foster, and T. Lyons, "Neural controlled differential equations for irregular time series," *arXiv preprint arXiv:2005.08926*, 2020.
[7] M. Horn, M. Moor, C. Bock, B. Rieck, and K. Borgwardt, "Set functions for time series," in *International Conference on Machine Learning*, pp. 4353–4363, PMLR, 2020.
[8] S. N. Shukla and B. M. Marlin, "Interpolation-prediction networks for irregularly sampled time series," *International Conference on Learning Representations*, 2019.
[9] V. K. Yalavarthi, J. Burchert, and L. Schmidt-Thieme, "DCSF: deep convolutional set functions for classification of asynchronous time series," in *9th IEEE International Conference on Data Science and Advanced Analytics, DSAA 2022, Shenzhen, China, October 13-16, 2022*, pp. 1–10, IEEE, 2022.
[10] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," *arXiv preprint arXiv:1806.07366*, 2018.
[11] E. V. Bonilla, K. Chai, and C. Williams, "Multi-task gaussian process prediction," *Advances in neural information processing systems*, vol. 20, 2007.
[12] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.
[13] R. Dürichen, M. A. Pimentel, L. Clifton, A. Schweikard, and D. A. Clifton, "Multitask gaussian processes for multivariate physiological time-series analysis," *IEEE Transactions on Biomedical Engineering*, vol. 62, no. 1, pp. 314–322, 2014.

TABLE X: Results for deterministic interpolation on all datasets. Evaluation measure is Mean Squared Error, lower the best.

tp. obs.	10%	50%	90%	10%	50%	90%
	random missing			burst missing		
	Physionet					
mTAN	0.530±0.001	0.377±0.001	0.368±0.002	0.646±0.001	0.569±0.002	0.392±0.001
HETVAE	0.549±0.001	0.378±0.001	0.343±0.002	0.635±0.002	0.567±0.002	**0.331±0.001**
Tripletformer	**0.525±0.006**	**0.372±0.007**	**0.331±0.011**	**0.627±0.001**	**0.554±0.004**	0.335±0.003
	MIMIC-III					
mTAN	0.661±0.001	0.477±0.001	0.474±0.001	0.805±0.001	0.710±0.001	0.511±0.001
HETVAE	0.676±0.001	0.472±0.001	0.430±0.001	0.804±0.002	0.727±0.002	**0.446±0.000**
Tripletformer	**0.646±0.023**	**0.462±0.017**	**0.414±0.008**	**0.788±0.002**	**0.684±0.001**	0.448±0.002
	Physionet2019					
mTAN	**0.581±0.002**	0.419±0.002	0.415±0.005	0.737±0.001	0.541±0.001	0.461±0.002
HETVAE	0.593±0.000	0.410±0.001	**0.350±0.000**	0.744±0.005	0.515±0.000	0.485±0.002
Tripletformer	0.585±0.001	**0.392±0.001**	0.371±0.003	**0.591±0.009**	**0.392±0.001**	**0.370±0.003**
	PenDigits					
mTAN	0.783±0.004	0.453±0.007	0.335±0.018	0.743±0.003	0.466±0.005	0.335±0.018
HETVAE	0.814±0.001	0.725±0.019	0.848±0.011	0.736±0.026	0.723±0.001	0.848±0.011
Tripletformer	**0.695±0.008**	**0.326±0.003**	**0.215±0.016**	**0.644±0.001**	**0.408±0.004**	**0.215±0.016**
	PhonemeSpectra					
mTAN	0.698±0.005	0.409±0.006	0.515±0.016	0.896±0.001	**0.782±0.002**	0.817±0.011
HETVAE	0.785±0.002	0.385±0.010	0.528±0.004	0.887±0.001	0.815±0.001	**0.808±0.002**
Tripletformer	**0.679±0.011**	**0.367±0.035**	**0.210±0.009**	**0.854±0.001**	0.808±0.001	0.796±0.002

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

[15] S. N. Shukla and B. M. Marlin, "Heteroscedastic temporal variational autoencoder for irregularly sampled time series," *International Conference on Learning Representation*, 2022.

[16] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh, "Set transformer: A framework for attention-based permutation-invariant neural networks," in *International conference on machine learning*, pp. 3744–3753, PMLR, 2019.

[17] J. Yoon, W. R. Zame, and M. Van Der Schaar, "Deep sensing: Active sensing using multi-directional recurrent neural networks," in *International Conference on Learning Representations*, 2018.

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

[19] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Scientific reports*, vol. 8, no. 1, p. 6085, 2018.

[20] X. Li, T.-K. L. Wong, R. T. Chen, and D. Duvenaud, "Scalable gradients for stochastic differential equations," in *International Conference on Artificial Intelligence and Statistics*, pp. 3870–3882, PMLR, 2020.

[21] A. Norcliffe, C. Bodnar, B. Day, J. Moss, and P. Liò, "Neural ode processes," *9th International Conference on Learning Representations, ICLR*, 2021.

[22] M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. Eslami, and Y. W. Teh, "Neural processes," *arXiv preprint arXiv:1807.01622*, 2018.

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

[24] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 11106–11115, 2021.

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

[26] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals," *circulation*, vol. 101, no. 23, pp. e215–e220, 2000.

[27] A. E. Johnson, T. J. Pollard, L. Shen, H. L. Li-Wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark, "Mimic-iii, a freely accessible critical care database," *Scientific data*, vol. 3, no. 1, pp. 1–9, 2016.

[28] H. Harutyunyan, H. Khachatrian, D. C. Kale, G. Ver Steeg, and A. Galstyan, "Multitask learning and benchmarking with clinical time series data," *Scientific data*, vol. 6, no. 1, pp. 1–18, 2019.

[29] M. A. Reyna, C. Josef, S. Seyedi, R. Jeter, S. P. Shashikumar, M. B. Westover, A. Sharma, S. Nemati, and G. D. Clifford, "Early prediction of sepsis from clinical data: the physionet/computing in cardiology challenge 2019," in *2019 Computing in Cardiology (CinC)*, pp. Page–1, IEEE, 2019.

[30] A. P. Ruiz, M. Flynn, J. Large, M. Middlehurst, and A. Bagnall, "The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances," *Data Mining and Knowledge Discovery*, vol. 35, no. 2, pp. 401–449, 2021.

APPENDIX

Hyperparameters searched for the competing models

GPR Following [15] we use squared exponential kernel. We search learning rate from $\{0.1, 0..1, 0.001\}$ and batch size from $\{32, 64, 128, 256\}$.

M-GPR In M-GPR we used convolution of kernels as shown in [13]. Similar to GPR, we search learning rate from $\{0.1, 0..1, 0.001\}$ and batch size from $\{32, 64, 128, 256\}$.

ODE models: We search the standard deviation over the range of $\{0.01, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0\}$, select the number of GRU hidden units, latent dimension, nodes in the fully connected network for the ode function in both encoder and decoder from $\{20, 32, 64, 128, 256\}$. Number of fully connected layers are searched in the range of $\{1, 2, 3\}$.

mTAN: We search the standard deviation from $\{0.01, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0\}$, number of attention heads from $\{1, 2, 4\}$, reference points from $\{8, 16, 32, 64, 128\}$, latent dimensions from $\{20, 30, 40, 50\}$, generator layers from $\{25, 50, 100, 150\}$, and reconstruction layers from $\{32, 64, 128, 256\}$.

HETVAE: We search the same hyperparameter range mention in [15]. We set time embedding dimension to 128, search hidden nodes in the decoder from $\{16, 32, 64, 128\}$, number reference points from $\{4, 8, 16, 32\}$, latent dimension from $\{8, 16, 32, 64, 128\}$, width of the fully connected layers from $\{128, 256, 512\}$ and augmented learning objective from $\{1.0, 5.0, 10.0\}$.