Self-Supervised Transformer for Sparse and Irregularly Sampled Multivariate Clinical Time-Series

SINDHU TIPIRNENI and CHANDAN K. REDDY, Virginia Tech

Multivariate time-series data are frequently observed in critical care settings and are typically characterized by sparsity (missing information) and irregular time intervals. Existing approaches for learning representations in this domain handle these challenges by either aggregation or imputation of values, which in-turn suppresses the fine-grained information and adds undesirable noise/overhead into the machine learning model. To tackle this problem, we propose a **S**elf-supervised **Tra**nsformer for **T**ime-**S**eries (STraTS) model, which overcomes these pitfalls by treating time-series as a set of observation triplets instead of using the standard dense matrix representation. It employs a novel Continuous Value Embedding technique to encode continuous time and variable values without the need for discretization. It is composed of a Transformer component with multi-head attention layers, which enable it to learn contextual triplet embeddings while avoiding the problems of recurrence and vanishing gradients that occur in recurrent architectures. In addition, to tackle the problem of limited availability of labeled data (which is typically observed in many healthcare applications), STraTS utilizes self-supervision by leveraging unlabeled data to learn better representations by using time-series forecasting as an auxiliary proxy task. Experiments on real-world multivariate clinical time-series benchmark datasets demonstrate that STraTS has better prediction performance than state-of-the-art methods for mortality prediction, especially when labeled data is limited. Finally, we also present an interpretable version of STraTS, which can identify important measurements in the time-series data. Our data preprocessing and model implementation codes are available at https://github.com/sindhura97/STraTS.

CCS Concepts: • **Computing methodologies → Neural networks**; **Transfer learning;**

Additional Key Words and Phrases: Time-series, neural networks, deep learning, healthcare, Transformer, self-supervised learning

ACM Reference format:
Sindhu Tipirneni and Chandan K. Reddy. 2022. Self-Supervised Transformer for Sparse and Irregularly Sampled Multivariate Clinical Time-Series. *ACM Trans. Knowl. Discov. Data.* 16, 6, Article 105 (July 2022), 17 pages.
https://doi.org/10.1145/3516367

1 INTRODUCTION

Time-series data is routinely collected in various healthcare settings where different measurements are recorded for patients throughout their course of stay (See Figure 1 for an illustrative example). Predicting clinical outcomes like mortality, decompensation, length of stay, and disease risk from such complex multivariate time-series data can facilitate both effective management of critical

This work was supported in part by the US National Science Foundation grants IIS-1838730 and Amazon AWS credits.
Authors' address: S. Tipirneni and C. K. Reddy, Virginia Tech, 900 N Glebe Road, Arlington, Virginia 22203; emails: tsaisindhura@vt.edu, reddy@cs.vt.edu.

1556-4681/2022/07-ART105
https://doi.org/10.1145/3516367

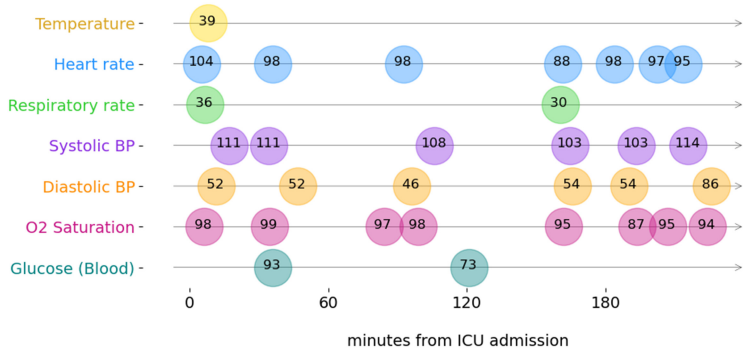

Fig. 1. An illustrative example of a multivariate clinical time-series with irregular time points and missing values.

care units and automatic personalized treatment recommendations for patients. The success of deep learning in image and text domains realized by convolutional and recurrent networks [7, 28], and Transformer models [30] have inspired the application of these architectures to develop better prediction models for time-series data as well. However, time-series in the clinical domain portray a unique set of challenges that are described below.

- **Missingness and Sparsity:** A patient's condition may demand observing only a subset of variables of interest. Thus, not all the variables are observed for every patient. Also, the observed time-series matrices are very sparse as some variables may be measured more frequently than others for a given patient.
- **Irregular time intervals and Sporadicity:** Not all clinical variables are measured at regular time intervals. Thus, the measurements may occur sporadically in time depending on the underlying condition of the patient.
- **Limited labeled data:** Patient-level clinical data is often expensive to obtain and labeled data subsets pertaining to a specific prediction task may be even more limited (for e.g., building a severity classifier for Covid-19 patients.)

A straight-forward approach to deal with irregular time intervals and missingness is to aggregate measurements into discrete time intervals and add missingness indicators, respectively. However, this suppresses important fine-grained information because the granularity of observed time-series may differ from patient to patient based on the underlying medical condition. Existing sequence models for clinical time-series [4] and other interpolation-based models [26] address this issue by including a learnable imputation or interpolation component. Such techniques add undesirable noise and extra overhead to the model, which usually worsens as the time-series become increasingly sparse. These models rely on an effective imputation/interpolation scheme in order to achieve strong performance on the target task. But it is unreasonable to impute clinical variables without careful consideration of the domain knowledge about each variable which might be non-trivial to obtain.

Considering these shortcomings, we design a framework that does not need to perform any such operations and directly builds a model *based only on the observations that are available in the data*. Thus, unlike conventional approaches, which view each time-series as a matrix of certain dimensions (#features × #time-steps), our model regards each time-series as a set of observation triplets (a triple containing time, variable, and value) without the necessity for aggregation or imputation. The proposed STraTS (acronym for **S**elf-supervised **Tra**nsformer for **T**ime-**S**eries) model embeds these triplets by using a novel **Continuous Value Embedding** (**CVE**) scheme to avoid

the need for binning continuous values before embedding them. The use of CVE for representing the time dimension preserves the fine grained information which is lost when the time-axis is discretized. STraTS encodes contextual information of observation triplets using a Transformer-based architecture with **multi-head attention** (**MHA**). We choose this over **recurrent neural network** (**RNN**) architectures because the sequential nature of RNN models hinders parallel processing while the Transformer bypasses this by using self-attention to attend from every token to every other token in a single step.

To build robust representations using limited labeled data, we employ self-supervision and develop a time-series forecasting task to pretrain STraTS. This enables learning generalized representations in the presence of limited labeled data and alleviates sensitivity to noise. Furthermore, interpretable models are usually preferred in healthcare but existing deep models for clinical time-series lack this component. Thus, we also propose an interpretable version of our model (I-STraTS) which slightly compromises on performance metrics but can identify important measurements in the input. Though we evaluate the proposed model only on binary classification tasks, our framework can also be utilized in other supervised and unsupervised settings, where learning robust and generalized representations of sparse and sporadic time-series is desired. The main contributions of our work can be summarized as follows:

— Propose a Transformer-based architecture called STraTS for clinical time-series, which addresses the unique challenges of missingness and sporadicity of such data by avoiding aggregation and imputation.
— Develop a novel CVE mechanism using a one-to-many **feed-forward network** (**FFN**) to embed continuous times and measured values in order to preserve fine grained information.
— Utilize forecasting as a self-supervision (proxy) task to leverage unlabeled data to learn more generalized and robust representations.
— Propose an interpretable version of STraTS that can be used when interpretability is more desired compared to quantitative performance gains.
— Demonstrate through an extensive set of experiments that the design choices of STraTS lead to a better performance compared to competitive baseline models for mortality prediction on two real-world clinical datasets.

The rest of this article is organized as follows. In Section 2, we review relevant literature about tackling sparse and sporadic time-series data, and self-supervised learning. Section 3 formally defines the prediction problem and gives a detailed description of the architecture of STraTS along with the self-supervision approach. Section 4 presents experimental results comparing STraTS with various baselines and demonstrates the interpretability of I-STraTS with a case study. Finally, Section 5 concludes the article and provides future directions.

2 RELATED WORK

2.1 Clinical Time-Series

A straightforward approach to address missing values and irregular time intervals is to impute and aggregate the time-series, respectively, before feeding them to a classifier [5, 20]. However, such classifiers ignore the missingness in the data, which can be quite informative. Lipton et al. [21] show that phenotyping performance can be improved by passing missingness indicators as additional features to an RNN classifier. But they still lose fine-grained information by aggregating each time-series into hourly intervals.

Several early works rely on **Gaussian Processes** (**GP**) [24] to model irregular time-series. For example, Lu et al. [23] represent each time-series as a smooth curve in a **reproducing kernel Hilbert space** (**RKHS**) using GP by optimizing GP parameters using **Expectation Maximization**

(**EM**), and then derive a distance measure on the RKHS, which is used to define the SVM classifier's kernel. To account for uncertainty in GP, Li and Marlin [18] formulate the kernel by applying an uncertainty-aware base kernel (called the expected Gaussian kernel) to a series of sliding windows. These works take a two-step approach by first optimizing GP parameters and then training the classification model. To enable end-to-end training, Li and Marlin [19] again represent time-series using GP posterior at predefined time points but use the reparametrization trick to back-propagate the gradients through a black-box classifier (learnable by gradient-descent) into the GP model. The end-to-end model is uncertainty-aware as the output is formulated as a random variable. Futoma et al. [11] extend this idea to multivariate time-series with the help of multitask GP [3] to consider inter-variable similarities. Though GP provide a systematic way to deal with uncertainty, they are expensive to learn and their flexibility is limited by choice of covariance and mean functions.

Shukla and Marlin [26] also propose an end-to-end method that constitutes interpolation and classification networks stacked in a sequence. They develop learnable interpolation layers to approximate the time-series at regular predefined time points in a deterministic fashion (unlike GP-based methods) and allow information sharing across both time and variable dimensions. However, the input to the classifier is a densely interpolated multivariate time-series which causes loss of information if the number of interpolation points is small and slows down computations while adding noise otherwise.

Instead of using a separate interpolation module followed by a traditional classifier, other approaches modify traditional recurrent architectures for clinical time-series to deal with missing values and/or irregular time intervals. For example, Baytas et al. [2] developed a **time-aware long-short term memory** (**T-LSTM**) which is a modification of the LSTM cell to adjust the hidden state according to the irregular time gaps. ODE-RNN [25] uses ODEs to model the continuous-time dynamics of the hidden state while also updating the hidden state at each observed time point using a standard GRU cell. The GRU-D model [4] is a modification of the GRU cell which decays inputs (to global means) and hidden states through unobserved time intervals. DATA-GRU [29], in addition to decaying the GRU hidden state according to elapsed time, also employs a dual attention mechanism based on missingness and imputation reliability to process inputs before feeding them to a GRU cell. All these methods use an RNN with sequence length being the number of unique timestamps in the input, which can be quite large for irregular time-series, and as a result, can slow down computations.

The imputation/interpolation schemes in the models discussed above can lead to excessive computations and unnecessary noise, particularly when missing rates are quite high. Our model is designed to circumvent this issue by representing sparse and irregular time-series as a set of observations. Horn et al. [13] develop SeFT with a similar idea and use a parametrized set function for classification. The attention-based aggregation used in SeFT contains the same queries for all observations to facilitate low memory and time complexity while compromising on accuracy. The initial embedding in SeFT contains fixed time encodings while our approach uses learnable embeddings for all the three components (time, variable, value) of the observation triplet.

The challenge of training in scenarios with limited labeled data still remains. In order to address this issue, we turn towards self-supervision for a better utilization of the available data to learn effective representations.

2.2 Self-Supervised Learning

Supervised deep learning models often rely on large amounts of labeled data to learn generalized and robust representations. Limited labeled data can make the model easily overfit to training data and make the model more sensitive to noise. Since labeled data is expensive to obtain, self-supervised learning was introduced as a technique to solve this challenge. This technique trains

the model on carefully constructed proxy tasks that improve the model's performance on target prediction tasks. The labeled datasets for proxy tasks are obtained from the unlabeled data in an inexpensive semi-automatic process. Yann Le Cunn[1] describes self-supervised learning as to "predict any part of the input from any other part". Self-supervised learning enables the model to learn correlations in input data which enhance the model's learning of supervised target prediction tasks. Liu et al. [22] review the state-of-the-art self-supervised learning methods in computer vision, natural language processing, and graph representation learning. Though this technique has shown great performance boosts with image [15] and text [9, 31] data, its application to time-series data has been limited. One such effort is made by Jawed et al. [14], which uses a 1D CNN for dense univariate time-series classification and shows increased accuracy by using forecasting as an additional task in a multi-task learning framework. Zerveas et al. [33] pretrained a Transformer model using a denoising objective and showed improved performance on regression and classification tasks with dense multivariate time-series. In our work, we demonstrate time-series forecasting as a viable and effective self-supervision task for a Transformer model. Our work is the first to explore self-supervised learning in the context of sparse and irregular multivariate time-series.

3 PROPOSED APPROACH

In this section, we describe our STraTS model by first introducing the problem with relevant notation and definitions and then explaining the different components of the model, which are illustrated in Figure 3.

3.1 Problem Definition

As stated in the previous sections, STraTS represents each time-series as a set of observation triplets. Formally, an *observation triplet* is defined as a triple (t, f, v) where $t \in \mathbb{R}_{\geq 0}$ is the time, $f \in \mathcal{F}$ is the feature/variable, and $v \in \mathbb{R}$ is the value of the observation. A *multivariate time-series* $\mathbf{T}$ of length n is a defined as a set of n observation triplets i.e., $\mathbf{T} = \{(t_i, f_i, v_i)\}_{i=1}^{n}$.

Consider a dataset $\mathcal{D} = \{(\mathbf{d}^k, \mathbf{T}^k, y^k)\}_{k=1}^{N}$ with N labeled samples, where the kth sample contains a demographic vector $\mathbf{d}^k \in \mathbb{R}^D$, a multivariate time-series $\mathbf{T}^k$, and a corresponding binary label $y^k \in \{0, 1\}$. In this work, each sample corresponds to a single ICU stay where several clinical variables of the patient are measured at irregular time intervals and the binary label indicates in-hospital mortality. The underlying set of time-series variables denoted by $\mathcal{F}$ may include vitals (such as temperature), lab measurements (such as hemoglobin), and input/output events (such as fluid intake and urine output). Thus, the *target task* aims at predicting y^k given $(\mathbf{d}^k, \mathbf{T}^k)$.

Our model also incorporates forecasting as a self-supervision task. For this task, we consider a bigger dataset with $N' \geq N$ samples given by $\mathcal{D}' = \{(\mathbf{d}^k, \mathbf{T}^k, \mathbf{m}^k, \mathbf{z}^k)\}_{k=1}^{N'}$. Here, $\mathbf{m}^k \in \{0, 1\}^{|\mathcal{F}|}$ is the forecast mask which indicates whether each variable was observed in the forecast window and $\mathbf{z}^k \in \mathbb{R}^{|\mathcal{F}|}$ contains the corresponding variable values when observed. The forecast mask is necessary because the unobserved forecasts cannot be used in training and are hence masked out in the loss function. The time-series in this dataset are obtained from both the labeled and unlabeled time-series by considering different observation windows. Figure 2 illustrates the construction of inputs and outputs for the target task and forecasting task.

3.2 Architecture of STraTS

The architecture of STraTS is illustrated in Figure 3. Unlike most of the existing approaches which take a time-series matrix as input, STraTS *defines its input as a set of observation triplets*. Each

[1] https://drive.google.com/file/d/1r-mDL4IX_hzZLDBKp8_e8VZqD7fOzBkF/view.

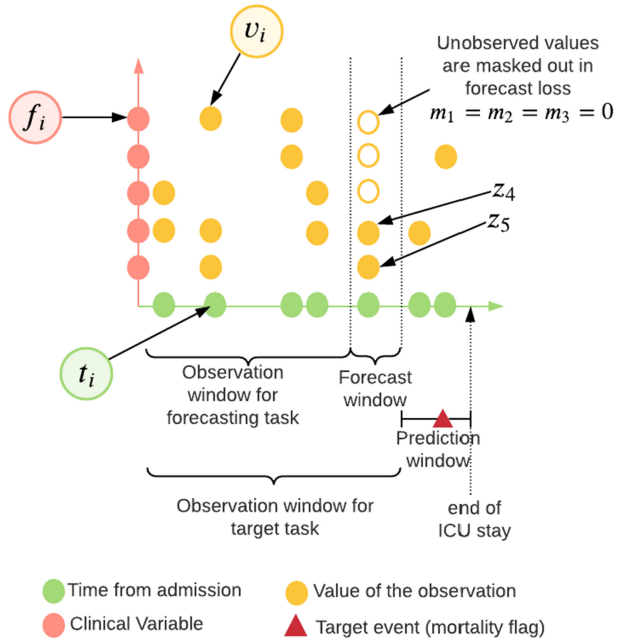

Fig. 2. An illustration of input and output construction for target and self-supervision (forecasting) tasks. The target task uses a fixed length observation window to predict in-hospital mortality. The forecasting task has an observation window that is followed by a fixed length prediction window in which only a subset of variables may be observed. Note that several observation windows are considered for each time-series for the forecasting task.

observation triplet in the input is embedded using the Initial Triplet Embedding module. The initial triplet embeddings are then passed through a Contextual Triplet Embedding module which utilizes the Transfomer architecture to encode the context for each triplet. The Fusion Self-attention module then combines these contextual embeddings via self-attention mechanism to generate an embedding for the input time-series which is concatenated with demographics embedding and passed through a FFN to make the final prediction. The notations used in the article are summarized in Table 1.

3.2.1 Initial Triplet Embedding. Given an input time-series $\mathbf{T} = \{(t_i, f_i, v_i)\}_{i=1}^{n}$, the initial embedding for the ith triplet $\mathbf{e_i} \in \mathbb{R}^d$ is computed by summing the following component embeddings: (i) *Feature embedding* $\mathbf{e_i^f} \in \mathbb{R}^d$, (ii) *Value embedding* $\mathbf{e_i^v} \in \mathbb{R}^d$, and (iii) *Time embedding* $\mathbf{e_i^t} \in \mathbb{R}^d$. In other words, $\mathbf{e_i} = \mathbf{e_i^f} + \mathbf{e_i^v} + \mathbf{e_i^t} \in \mathbb{R}^d$. Feature embeddings $\mathbf{e^f}(\cdot)$ are obtained from a simple lookup table similar to word embeddings. Since feature values and times are continuous unlike feature names which are categorical objects, we cannot use a lookup table to embed these continuous values unless they are categorized. Some researchers [30, 32] have used sinusoidal encodings to embed continuous values. We propose a novel CVE technique using a one-to-many FFN with learnable parameters i.e., $\mathbf{e_i^v} = FFN^v(v_i)$, and $\mathbf{e_i^t} = FFN^t(t_i)$.

Both FFNs have one input neuron and d output neurons and a single hidden layer with $\lfloor \sqrt{d} \rfloor$ neurons and $tanh(\cdot)$ activation. They are of the form $FFN(x) = U\, tanh(Wx + b)$ where the dimensions of weights $\{W, b, U\}$ can be inferred from the size of hidden and output layers of the FFN. Unlike sinusoidal encodings with fixed frequencies, this technique offers more flexibility by allowing end-to-end learning of continuous value and time embeddings without the need to categorize them.

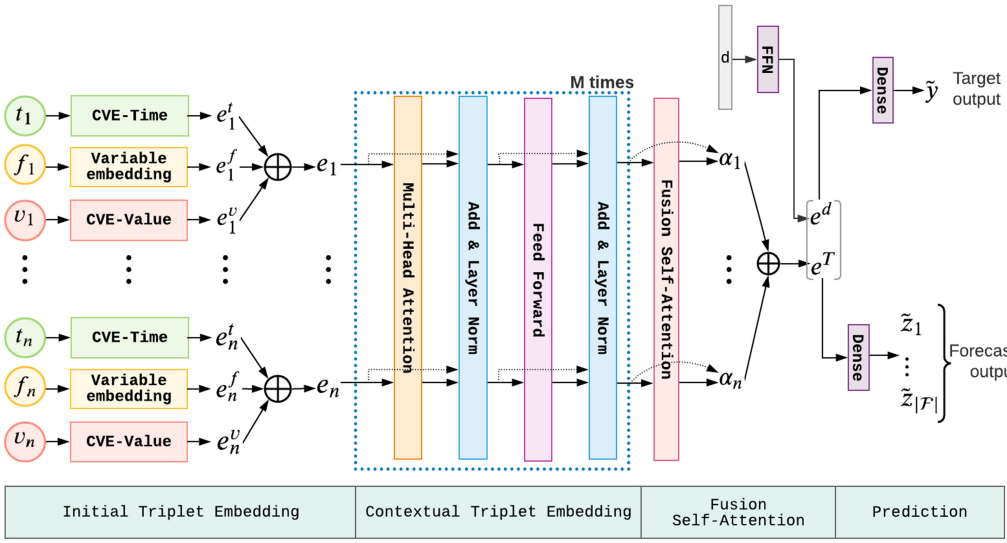

Fig. 3. The overall architecture of the proposed STraTS model. The Input Triplet Embedding module embeds each observation triplet, the Contextual Triplet Embedding module encodes contextual information for the triplets, the Fusion Self-Attention module computes time-series embedding, which is concatenated with demographics embedding and passed through a dense layer to generate predictions for target and self-supervision (forecasting) tasks.

Table 1. Notations used in this Article

Notation	Definition
N	# Time-series for target task
N'	# Time-series for forecasting task
$\mathbf{d} \in \mathbb{R}^D$	Demographics vector
$\mathcal{F}$	Set of clinical variables
$t_i \in \mathbb{R}_{\geq 0}$	Time of ith observation
$f_i \in \mathcal{F}$	Variable of ith observation
$v_i \in \mathbb{R}$	Value of ith observation
(t, f, v)	Observation triplet
$\mathbf{T} = \{(t_i, f_i, v_i)\}_{i=1}^n$	Multivariate time-series
$y, \tilde{y} \in \{0, 1\}$	True and predicted outputs for target task
$\mathbf{z}, \tilde{\mathbf{z}} \in \mathbb{R}^{\lvert\mathcal{F}\rvert}$	True and predicted outputs for forecasting task
$\mathbf{m} \in \{0, 1\}^{\lvert\mathcal{F}\rvert}$	Forecast mask
$\mathbf{e}_i^t, \mathbf{e}_i^v \in \mathbb{R}^d$	CVE for time and value
$\mathbf{e}_i^f \in \mathbb{R}^d$	Variable embedding
$\mathbf{e}_i \in \mathbb{R}^d$	Initial triplet embedding
$\mathbf{e}^T \in \mathbb{R}^d$	Time-series embedding
$\mathbf{e}^d \in \mathbb{R}^d$	Demographics embedding

3.2.2 Contextual Triplet Embedding. The initial triplet embeddings $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$ are then passed through a Transformer architecture [30] with M blocks, each containing a MHA layer with h attention heads and a FFN with one hidden layer. Each block takes n input embeddings $\mathbf{E} \in \mathbb{R}^{n \times d}$ and outputs the corresponding n output embeddings $\mathbf{C} \in \mathbb{R}^{n \times d}$ that capture the contextual

information. MHA layers use multiple attention heads to attend to information contained in different embedding projections in parallel. The computations of the MHA layer are given by

$$\mathbf{H}_j = softmax\left(\frac{\left(\mathbf{E}\mathbf{W}_j^q\right)\left(\mathbf{E}\mathbf{W}_j^k\right)^T}{\sqrt{d/h}}\right)\left(\mathbf{E}\mathbf{W}_j^v\right) \quad j = 1, \ldots, h, \tag{1}$$

$$MHA(\mathbf{E}) = (\mathbf{H}_1 \circ \cdots \circ \mathbf{H}_h)\,\mathbf{W}_c. \tag{2}$$

Each head projects the input embeddings into query, key, and value subspaces using matrices $\{\mathbf{W}_j^q, \mathbf{W}_j^k, \mathbf{W}_j^v\} \subset \mathbb{R}^{d \times d_h}$. The queries and keys are then used to compute the attention weights which are used to compute weighted averages of value (different from value in observation triplet) vectors. Finally, the outputs of all heads are concatenated and projected to original dimension with $\mathbf{W}_c \in \mathbb{R}^{hd_h \times d}$. The FFN layer takes the form

$$\mathbf{F}(\mathbf{X}) = ReLU\left(\mathbf{X}\mathbf{W}_1^f + \mathbf{b}_1^f\right)\mathbf{W}_2^f + \mathbf{b}_2^f, \tag{3}$$

with weights $\mathbf{W}_1^{\mathbf{f}} \in \mathbb{R}^{d \times 2d}$, $\mathbf{b}_1^{\mathbf{f}} \in \mathbb{R}^{2d}$, $\mathbf{W}_2^{\mathbf{f}} \in \mathbb{R}^{2d \times d}$, $\mathbf{b}_2^{\mathbf{f}} \in \mathbb{R}^d$. Dropout, residual connections, and layer normalization are added for every MHA and FFN layer. Also, attention dropout randomly masks out some positions in the attention matrix before the softmax computation during training. The output of each block is fed as input to the succeeding one, and the output of the last block gives the contextual triplet embeddings $\{\mathbf{c}_1, \ldots, \mathbf{c}_n\}$.

3.2.3 Fusion Self-Attention. After computing contextual embeddings using a Transformer, we fuse them using a self-attention layer to compute time-series embedding $\mathbf{e}^T \in \mathbb{R}^d$. This layer first computes attention weights $\{\alpha_1, \ldots, \alpha_n\}$ by passing each contextual embedding through a FFN and computing a softmax over all the FFN outputs.

$$a_i = \mathbf{u}_a^T\, tanh(\mathbf{W}_a\mathbf{c}_i + \mathbf{b}_a), \tag{4}$$

$$\alpha_i = \frac{exp(a_i)}{\sum_{j=1}^n exp(a_j)} \quad \forall i = 1, \ldots, n, \tag{5}$$

$\mathbf{W}_a \in \mathbb{R}^{d_a \times d}, \mathbf{b}_a \in \mathbb{R}^{d_a}, \mathbf{u}_\mathbf{a} \in \mathbb{R}^{d_a}$ are the weights of this attention network which has d_a neurons in the hidden layer. The time-series embedding is then computed as

$$\mathbf{e}^T = \sum_{i=1}^n \alpha_i \mathbf{c}_\mathbf{i}. \tag{6}$$

3.2.4 Demographics Embedding. We realize that demographics can be encoded as triplets with a default value for time. However, we found that the prediction models performed better in our experiments when demographics are processed separately by passing $\mathbf{d}$ through a FFN as shown below. The demographics embedding is thus obtained as

$$\mathbf{e}^d = tanh\left(\mathbf{W}_2^d\, tanh\left(\mathbf{W}_1^d\mathbf{d} + \mathbf{b}_1^d\right) + \mathbf{b}_2^d\right) \in \mathbb{R}^d, \tag{7}$$

where the hidden layer has a dimension of $2d$.

3.2.5 Prediction Head. The final prediction for target task is obtained by passing the concatenation of demographics and time-series embeddings through a dense layer with weights $\mathbf{w}_o^T \in \mathbb{R}^d$, $b_o \in \mathbb{R}$ and sigmoid activation.

$$\tilde{y} = sigmoid\left(\mathbf{w}_o^T[\mathbf{e}^d \circ \mathbf{e}^T] + b_o\right). \tag{8}$$

The model is trained on the target task using cross-entropy loss.

3.2.6 Self-Supervision. We experimented with both masking and forecasting as pretext tasks for providing self-supervision and found that forecasting improved the results on target tasks. The forecasting task uses the same architecture as the target task except for the prediction layer i.e.,

$$\tilde{\mathbf{z}} = \mathbf{W}_s[\mathbf{e}^d \circ \mathbf{e}^T] + \mathbf{b}_s \in \mathbb{R}^{|\mathcal{F}|}. \tag{9}$$

A masked MSE loss is used for training on the forecasting task to account for missing values in the forecast outputs. Thus, the loss for self-supervision is given by

$$\mathcal{L}_{ss} = \frac{1}{|N'|} \sum_{k=1}^{N'} \sum_{j=1}^{|\mathcal{F}|} m_j^k \left(\tilde{\mathbf{z}}_j^k - \mathbf{z}_j^k\right)^2, \tag{10}$$

where $\mathbf{m}_j^k = 1$ (or $\mathbf{m}_j^k = 0$) if the ground truth forecast $\mathbf{z}_j^k$ is available (or unavailable) for jth variable in kth sample. The model is first pretrained on the self-supervision task and is then fine-tuned on the target task.

3.3 Interpretability

We also propose an interpretable version of our model, which we refer to as I-STraTS. Inspired by Choi et al. [6] and Zhang et al. [34], we alter the architecture of STraTS in such a way that the output can be expressed using a linear combination of components that are derived from individual features. Specifically, the output of I-STraTS is formulated as

$$\tilde{y} = sigmoid\left(\mathbf{w}_o^T\left[\mathbf{d} \circ \sum_{i=1}^{n} \alpha_i \mathbf{e}_i\right] + b_o\right). \tag{11}$$

Contrary to STraTS, (i) we combine the initial triplet embeddings using the attention weights in Fusion Self-attention module, and (ii) directly use the raw demographics vector as the demographics embedding. The above equation can also be written as

$$\tilde{y} = sigmoid\left(\sum_{j=1}^{D} \mathbf{w}_o[j]\,\mathbf{d}[j] + \sum_{i=1}^{n}\sum_{j=1}^{d} \alpha_i\, \mathbf{w}_o[j+D]\,\mathbf{e}_i[j] + b_o\right) \tag{12}$$

Thus, we assign a "contribution score" to the jth demographic feature as $\mathbf{w}_o[j]\mathbf{d}[j]$ and to the ith time-series observation as $\sum_{j=1}^{d} \alpha_i\, \mathbf{w}_o[j+D]\,\mathbf{e}_i[j]$.

4 EXPERIMENTS

We evaluated our proposed STraTS model against state-of-the-art baselines on two real-world EHR databases for the mortality prediction task. This section starts with a description of the datasets and baselines, followed by a discussion of results focusing on generalization and interpretability.

4.1 Datasets

We experiment with time-series extracted from two real-world EHR datasets, which are described below. The dataset statistics are summarized in Table 2.

MIMIC-III [16]**:** This is a publicly available database containing medical records of about 46k critical care patients in Beth Israel Deaconess Medical Center between 2001 and 2012. We filtered ICU stays to include only adult patients and extracted 129 features from the following tables: input events, output events, lab events, chart events, and prescriptions for each ICU stay. For mortality prediction task, we only include ICU stays that lasted for atleast one day with the patient alive at the end of first day, and predict in-hospital mortality using the first 24 hours of data. For forecasting,

Table 2. Basic Statistics of the Two Datasets used in our Experiemnts

	MIMIC-III	PhysioNet-2012
# ICU stays	52,871	11,988
# ICU stays (supervised)	44,812	11,988
# Avg. span of time-series	101.9h	47.3h
# Avg. span of time-sries (supervised)	23.5h	47.3h
# Variables	129	37
Avg. variable missing rate	89.7%	79.7%
Avg. # observations/stay	401	436
Demographics	Age, Gender	Age, Gender, Height, ICU Type
Task	24-hour mortality	48-hour mortality
% positive class	9.7%	14.2%

Note that the Avg. variable missing rate and Avg. # observations/stay are calculated using only the supervised samples.

the set of observation windows is defined (in hours) as $\{[min(0, x-24), x) \mid 20 \leq x \leq 124,\ x\%4 = 0\}$ and the prediction window is the 2-hour period following the observation window. Note that we only consider those samples which have atleast one time-series measurement in both observation and prediction windows. The data is split at patient level into training, validation, and test sets in the ratio $64 : 16 : 20$.

PhysioNet Challenge 2012 [12]**:** This processed dataset from Physionet Challenge 2012[2] contains records of 11,988 ICU stays of adult patients. The target task aims at predicting in-hospital mortality given the first 48 hours of data for each ICU stay. Since demographic variables "gender" and "height" are not available for all ICU stays, we perform mean imputation and add missingness indicators for them as additional demographic variables. To generate inputs and outputs for forecasting, the set of observation windows is defined (in hours) as $\{[0, x) \mid 12 \leq x \leq 44,\ x\%4 = 0\}$ and the prediction window is the 2-hour period following the observation window. The data from set-b and set-c together is split into training and validation (80:20) while set-a is used for testing.

4.2 Baseline Methods

To demonstrate the effectiveness of STraTS over the state-of-the-art methods, we compare it with the following baseline models.

- **Gated Recurrent Unit (GRU)** [7]**:** The input is a time-series matrix with hourly aggregation where missing variables are mean-imputed. Binary missingness indicators and time since the last observation of each variable are also included as additional features at each time step. The final hidden state is transformed by a dense layer to generate output.
- **Temporal Convolutional Network (TCN)** [1]**:** This model takes the same input as GRU which is passed through a stack of temporal convolution layers with residual connections. The representation from the last time step of the last layer is transformed by a dense layer to generate output.
- **Simply Attend and Diagnose (SaND)** [27]**:** This model also has the same input representation as GRU and the input is passed through a Transformer with causal attention and a dense interpolation layer.
- **GRU with trainable Decays (GRU-D)** [4]**:** The GRU-D cell takes a vector of variable values at each time one or more measurements are seen. The GRU-D cell, which is a modification to

[2] https://physionet.org/content/challenge-2012/1.0.0/.

the GRU cell, decays unobserved values in this vector to global mean values and also adjusts the hidden state according to elapsed times since the last observation of each variable.

— **Interpolation-prediction Network (InterpNet)** [26]**:** This model consists of a semi-parametric interpolation network that interpolates all variables at regular predefined time points, followed by a prediction network which is a GRU. It also uses a reconstruction loss to enhance the interpolation network. The input representation is similar to that of GRU-D and therefore, no aggregation is performed.

— **Set Functions for Time Series (SeFT)** [13]**:** This model also inputs a set of observation triplets, similar to STraTS. It uses sinusoidal encodings to embed times and the deep network used to combine the observation embeddings is formulated as a set function using a simpler but faster variation of MHA.

For all the baselines, we use two dense layers to get the demographics encoding and concatenate it to the time-series representation before the last dense layer. All the baselines use sigmoid activation at the last dense layer for mortality prediction. The time-series measurements (by variable) and demographics vectors are normalized to have zero mean and unit variance. All models are trained using the Adam optimizer [17].

4.3 Evaluation Metrics

The following metrics are used to quantitatively compare the baselines and proposed models for the binary classification task of mortality prediction. (i) ROC-AUC: Area under ROC curve. (ii) PR-AUC: Area under precision-recall curve. (iii) min(Re, Pr): This metric is computed as the maximum of "minimum of recall and precision" across all thresholds.

4.4 Implementation Details

Table 3 lists the hyperparameters used in the experiments for all models for MIMIC-III and PhysioNet-2012 datasets. All models are trained using a batch size of 32 with Adam optimizer and training is stopped when sum of ROC-AUC and PR-AUC does not improve for 10 epochs. For pretraining phase using the self-supervision task, the patience is set to 5 epochs and epoch size is set to 256,000 samples. For MIMIC-III dataset, we set the maximum number of time-steps for GRU-D and InterpNet, and the maximum no. of observations for STraTS using the 99th percentile for the same. This is done to avoid memory overflow with batch gradient descent. The deep models are implemented using keras with tensorflow backend. For InterpNet, we adapted the official code from https://github.com/mlds-lab/interp-net. For GRU-D and SeFT, we borrowed implementations from https://github.com/BorgwardtLab/Set_Functions_for_Time_Series. The experiments are conducted on a single NVIDIA GRID P40-12Q GPU. Our implementation and data-processing codes for STraTS are available at https://github.com/sindhura97/STraTS.

4.5 Prediction Performance

We train each model using 10 different random samplings of 50% labeled data from the train and validation sets. Note that STraTS uses the entire labeled data and additional unlabeled data (if available) for self-supervision. Table 4 shows the results for mortality prediction on MIMIC-III and PhysioNet-2012 datasets which are averaged over the 10 runs. STraTS achieves the best performance on all metrics, improving PR-AUC by 3.2% and 3.5% on MIMIC-III and PhysioNet-2012 datasets over the best baseline, respectively. This shows that our design choices of triplet embedding, attention-based architecture, and self-supervision enable STraTS to learn better representations. We expected the interpolation-based models GRU-D and InterpNet to outperform the simpler models GRU, TCN, and SaND. This was true for all cases except that GRU showed a better performance than GRU-D and InterpNet on the MIMIC-III dataset, for reasons that are unclear.

Table 3. Hyperparameters used for Our Experiments in this Article

Model	MIMIC-III	PhysioNet-2012
GRU	units = 50, rec d/o = 0.2, output d/o = 0.2, lr = 0.0001	units = 43, rec d/o = 0.2, output d/o = 0.2, lr = 0.0001
TCN	layers = 4, filters = 128, kernel size = 4, d/o = 0.1, lr = 0.0001	layers = 6, filters = 64, kernel size = 4, d/o = 0.1, lr = 0.0005
SAnD	N = 4, r = 24, M = 12, d/o = 0.3, d = 64, h = 2, he = 8, lr = 0.0005	N = 4, r = 24, M = 12, d/o = 0.3, d = 64, h = 2, he = 8, lr = 0.0005
GRU-D	units = 60, rec d/o = 0.2, output d/o = 0.2, lr = 0.0001	units = 49 rec d/o = 0.2, output d/o = 0.2, lr = 0.0001
SeFT	lr = 0.001, 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 = 100.0, n_positional_dims = 4	lr = 0.00081, 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 = 100.0, n_positional_dims = 4
InterpNet	ref_points = 96, units = 100, input d/o = 0.2, rec d/o = 0.2, lr = 0.001	ref_points = 192, units = 100, input d/o = 0.2, rec d/o = 0.2, lr = 0.001
STraTS(ss-) & I-STraTS(ss-)	d = 32, M = 2, h = 4, d/o = 0.2, lr = 0.0005	d = 32, M = 2, h = 4, d/o = 0.2, lr = 0.001
STraTS & I-STraTS	d = 50, M = 2, h = 4, d/o = 0.2, lr = 0.0005	d = 50, M = 2, h = 4, d/o = 0.2, lr = 0.0005

Table 4. Mortality Prediction Performance on MIMIC-III and PhysioNet-2012 Datasets

		ROC-AUC	PR-AUC	min(Re,Pr)
MIMIC-III	GRU	0.886 ± 0.002	0.559 ± 0.005	0.533 ± 0.007
	TCN	0.879 ± 0.001	0.540 ± 0.004	0.525 ± 0.005
	SAnD	0.876 ± 0.002	0.533 ± 0.011	0.515 ± 0.008
	GRU-D	0.883 ± 0.003	0.544 ± 0.007	0.527 ± 0.005
	InterpNet	0.881 ± 0.002	0.540 ± 0.007	0.516 ± 0.005
	SeFT	0.881 ± 0.003	0.547 ± 0.011	0.524 ± 0.01
	STraTS	**0.891 ± 0.002**	**0.577 ± 0.006**	**0.541 ± 0.008**
PhysioNet-2012	GRU	0.831 ± 0.003	0.468 ± 0.008	0.465 ± 0.009
	TCN	0.813 ± 0.005	0.430 ± 0.01	0.433 ± 0.009
	SAnD	0.800 ± 0.013	0.406 ± 0.021	0.418 ± 0.018
	GRU-D	0.833 ± 0.005	0.481 ± 0.008	0.468 ± 0.012
	InterpNet	0.822 ± 0.007	0.460 ± 0.017	0.455 ± 0.017
	SeFT	0.832 ± 0.005	0.454 ± 0.017	0.465 ± 0.009
	STraTS	**0.839 ± 0.008**	**0.498 ± 0.012**	**0.483 ± 0.01**

The results show mean and standard deviation of the metrics after repeating the experiment 10 times by sampling 50% labeled data each time.

To test the generalization ability of different models, we evaluate STraTS and the baseline models by training them on varying percentages of labeled data. Lower proportions of labeled data can be observed in real-world when there are several right-censored samples. Figures 4 and 5 show the results for MIMIC-III and PhysioNet-2012 datasets, respectively. The performance of

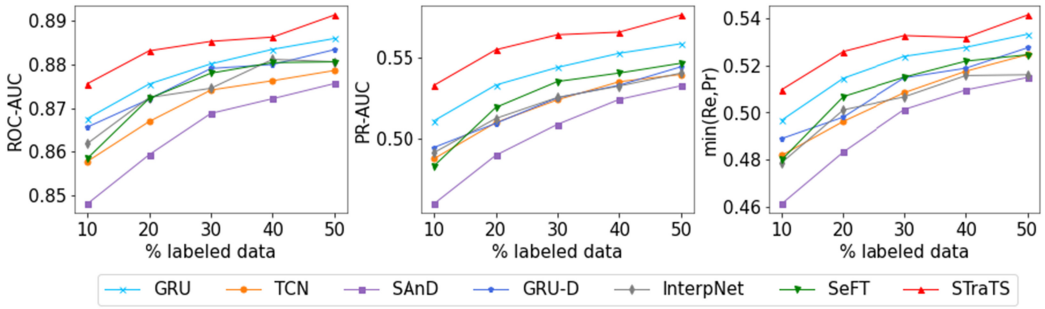

Fig. 4. Mortality prediction performance on MIMIC-III dataset for different percentages of labeled data averaged over 10 runs.

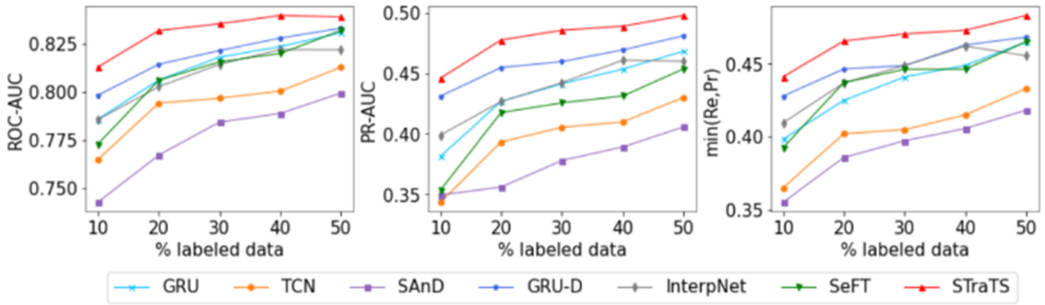

Fig. 5. Mortality prediction performance on PhysioNet-2012 dataset for different percentages of labeled data averaged over 10 runs.

all models degrades with reduced amount of labeled data. But STraTS is seen to have a crucial advantage compared to other models in scarce labeled data settings, which can be attributed to self-supervision.

4.6 Ablation Study

We compared the predictive performance of STraTs and I-STraTS, with and without self-supervision and the results are reported in Table 5. "ss+" and "ss−" are used to refer to models trained with and without self-supervision, respectively. We observe that (i) Adding interpretability to STraTS slightly reduces the prediction scores as a result of constraining model representations. (ii) Adding self-supervision improves performance of both STraTS and I-STraTS. (iii) I-STraTS(ss+) outperforms STraTS(ss−) on all metrics on MIMIC-III dataset, and on the PR-AUC metric for PhysioNet-2012 dataset. This demonstrates that the performance drop from introducing interpretability can be compensated by the performance improvements obtained through self-supervision.

4.7 Interpretability

To illustrate how I-STraTS explains its predictions, we present a case study for an 85-year-old female patient from the MIMIC-III dataset who expired on the 6th day after ICU admission. The I-STraTS model predicts the probability of her in-hospital mortality as 0.94 using only the data collected on the first day. The patient had 380 measurements corresponding to 58 time-series variables. The top 5 variables ordered by their average "contribution score" along with the range (for

Table 5. Ablation Study: Comparing Mortality Prediction Performance of STraTS and I-STraTS with and Without Self-Supervision

		ROC-AUC	PR-AUC	min(Re,Pr)
MIMIC-III	I-STraTS (ss−)	0.878 ± 0.002	0.542 ± 0.006	0.516 ± 0.008
	I-STraTS (ss+)	0.887 ± 0.003	0.556 ± 0.008	0.531 ± 0.005
	STraTS (ss−)	0.881 ± 0.002	0.546 ± 0.007	0.525 ± 0.012
	STraTS (ss+)	**0.891 ± 0.002**	**0.577 ± 0.006**	**0.541 ± 0.008**
PhysioNet-2012	I-STraTS (ss−)	0.826 ± 0.008	0.456 ± 0.018	0.467 ± 0.025
	I-STraTS (ss+)	0.833 ± 0.007	0.478 ± 0.015	0.466 ± 0.010
	STraTS (ss−)	0.835 ± 0.009	0.467 ± 0.023	0.471 ± 0.017
	STraTS (ss+)	**0.839 ± 0.008**	**0.498 ± 0.012**	**0.483 ± 0.010**

('ss+' and 'ss−' are used to indicate models trained with and without self-supervision, respectively.)

Table 6. Case Study: Top 5 Variables Ordered by 'average contribution score' Obtained from I-STraTS Model for an ICU Stay from MIMIC-III Dataset

Variable	Range/Value	'avg. contribution score'
Age	85	0.458
Lactate	[1.7, 6.4] mmol/L	0.175
LDH	[275, 306] IU/L	0.115
Platelet Count	[127, 132] K/uL	0.100
RDW	[22.0–22.1]%	0.083

multiple observations) or value (for only one observation) are shown in Table 6. In addition to old age, we can also observe that I-STraTS considers the abnormal values of Lactate, LDH, Platelet count, and RDW as the most important factors in predicting that the patient is at high risk of mortality. The discharge summary for this patient indicates PEA arrest as the cause of death. Elevated Lactate and LDH levels as seen in this case are known to be associated with cardiac arrest [8, 10]. Such predictions cannot only guide the care givers in identifying high-risk patients for better resource allocation but also guide the clinicians into understanding the contributing factors and make better diagnoses and treatment choices, especially at the early stages of treatment before the condition becomes more severe and uncontrollable.

To obtain a more fine-grained intuition, the observed time-series for some variables in this ICU stay are plotted in Figure 6, along with the corresponding contribution scores. It is interesting to see that the contribution scores appear to be positively or negatively correlated with the underlying values or time for several variables. For example, the model gives more weight to higher values of Lactate and LDH that are linked to cardiac arrest, which is the patient's cause of death. Similarly, the model pays more attention to increased blood glucose of 210 mg/dL. As GCS-verbal remains at a constant low of 1, the model gives it more and more weight as time progresses.

5 CONCLUSION

We proposed a Transformer-based model, STraTS, for prediction tasks on multivariate clinical time-series to address the challenges faced by existing methods in this domain. Our approach of using observation triplets as time-series components avoids the problems faced by aggregation and imputation methods for sparse and sporadic multivariate time-series. We used a novel CVE technique which uses parameterized embeddings for continuous values and multi-head attention layers to learn contextual representations. The self-supervision task of forecasting using unlabeled data enables STraTS to learn more generalized representations, thus outperforming state-of-the-art

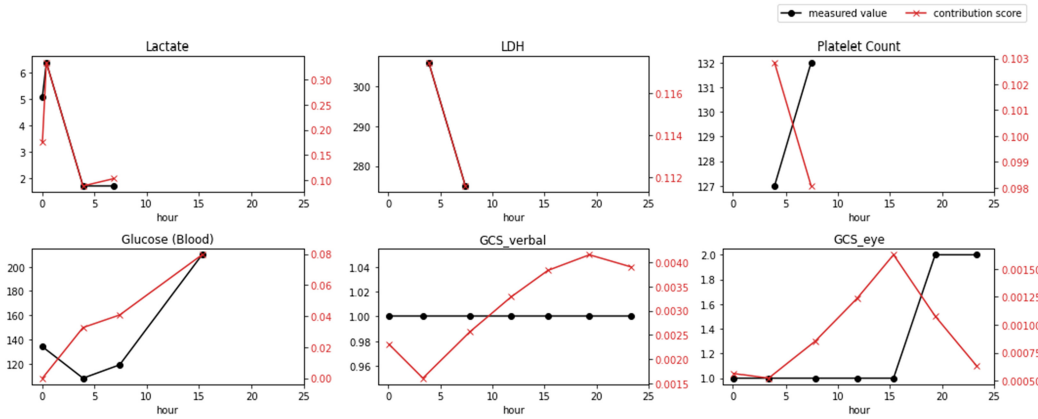

Fig. 6. Case study: An illustration of a few time-series with contribution scores for a patient from MIMIC-III dataset.

baselines. In addition, we also showed that STraTS generalizes well even when labeled data is scarce and is also more robust to noise compared to existing methods. We also proposed an interpretable version of STraTS, called I-STraTS, for which self-supervision compensates the drop in prediction performance from introducing interpretability. This work can motivate other researchers to explore more self-supervision tasks for clinical time-series data. Along with exploring more self-supervision tasks, future work should look at adapting STraTS or optimizing its computational efficiency for longer time series where attention matrices can become large and infeasible.

ACKNOWLEDGMENTS

We thank Lakshmi Tipirneni for her help with the clinical domain knowledge related to the extraction of our time-series dataset from MIMIC-III database and for providing clinical insights on the case study presented in Section 4.7.

REFERENCES

[1] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2018. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. CoRR abs/1803.01271 (2018). arXiv:1803.01271 http://arxiv.org/abs/1803.01271

[2] Inci M. Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K. Jain, and Jiayu Zhou. 2017. Patient subtyping via time-aware LSTM networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax*. ACM, 65–74. DOI: https://doi.org/10.1145/3097983.3097997

[3] Edwin V. Bonilla, Kian Ming Adam Chai, and Christopher K. I. Williams. 2007. Multi-task gaussian process prediction. In *Proceedings of the 21st Annual Conference on Advances in Neural Information Processing Systems*. Curran Associates, Inc., 153–160.

[4] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. 2018. Recurrent neural networks for multivariate time series with missing values. *Scientific Reports* 8, 1 (2018), 1–12. DOI: https://doi.org/10.1038/s41598-018-24271-9

[5] Weitong Chen, Sen Wang, Guodong Long, Lina Yao, Quan Z. Sheng, and Xue Li. 2018. Dynamic illness severity prediction via multi-task RNNs for intensive care unit. In *Proceedings of the IEEE International Conference on Data Mining, ICDM 2018*. IEEE Computer Society, 917–922. DOI: https://doi.org/10.1109/ICDM.2018.00111

[6] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter F. Stewart. 2016. RETAIN: An interpretable predictive model for healthcare using reverse time attention mechanism. In *Proceedings of the Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems*. 3504–3512.

[7] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. CoRR abs/1412.3555 (2014). arXiv:1412.3555 http://arxiv.org/abs/1412.3555

[8] Antonio Maria Dell'Anna, Claudio Sandroni, Irene Lamanna, Ilaria Belloni, Katia Donadello, Jacques Creteur, Jean-Louis Vincent, and Fabio Silvio Taccone. 2017. Prognostic implications of blood lactate concentrations after cardiac arrest: a retrospective study. *Annals of Intensive Care* 7, 1 (2017), 1–9.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the NAACL-HLT,* Vol 1. 4171–4186. DOI: https://doi.org/10.18653/v1/n19-1423

[10] A. Farhana and S. L. Lappin. 2021. Biochemistry, Lactate Dehydrogenase.[Updated 2020 May 17]. *StatPearls [Internet]. Treasure Island (FL): StatPearls Publishing* (2021).

[11] Joseph Futoma, Sanjay Hariharan, and Katherine A. Heller. 2017. Learning to detect sepsis with a multitask gaussian process RNN classifier. In *Proceedings of the 34th International Conference on Machine Learning, (ICML'17)* . PMLR, 1174–1182.

[12] Ary L. Goldberger, Luis A. N. Amaral, Leon Glass, Jeffrey M. Hausdorff, Plamen Ch Ivanov, Roger G. Mark, Joseph E. Mietus, George B. Moody, Chung-Kang Peng, and H. Eugene Stanley. 2000. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation* 101, 23 (2000), e215–e220.

[13] Max Horn, Michael Moor, Christian Bock, Bastian Rieck, and Karsten M. Borgwardt. 2020. Set functions for time series. In *Proceedings of the 37th International Conference on Machine Learning, (ICML'20)*, Vol. 119. PMLR, 4353–4363.

[14] Shayan Jawed, Josif Grabocka, and Lars Schmidt-Thieme. 2020. Self-supervised learning for semi-supervised time series classification. In *Proceedings of the Advances in Knowledge Discovery and Data Mining - 24th Pacific-Asia Conference, PAKDD 2020, Part I (Lecture Notes in Computer Science, Vol. 12084)*. Springer, 499–511. DOI: https://doi.org/10.1007/978-3-030-47426-3_39

[15] Longlong Jing and Yingli Tian. 2021. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43, 11 (2021), 4037–4058. DOI: https://doi.org/10.1109/TPAMI.2020.2992393

[16] Alistair E. W. Johnson, Tom J. Pollard, Lu Shen, H. Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific Data* 3 (2016), 160035. DOI: https://doi.org/10.1038/sdata.2016.35

[17] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations, (ICLR'15)*.

[18] Steven Cheng-Xian Li and Benjamin M. Marlin. 2015. Classification of sparse and irregularly sampled time series with mixtures of expected gaussian kernels and random features. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 484–493.

[19] Steven Cheng-Xian Li and Benjamin M. Marlin. 2016. A scalable end-to-end Gaussian process adapter for irregularly sampled time series classification. In *Proceedings of the Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems* (2016). 1804–1812.

[20] Zachary Chase Lipton, David C. Kale, and Randall C. Wetzel. 2015. Phenotyping of Clinical Time Series with LSTM Recurrent Neural Networks. CoRR abs/1510.07641 (2015). arXiv:1510.07641 http://arxiv.org/abs/1510.07641

[21] Zachary C. Lipton, David C. Kale, and Randall C. Wetzel. 2016. Directly modeling missing data in sequences with RNNs: Improved classification of clinical time series. In *Proceedings of the 1st Machine Learning in Health Care, MLHC 2016 (JMLR Workshop and Conference Proceedings, Vol. 56)*. JMLR.org, 253–270.

[22] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang. 2021. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge & Data Engineering* 1 (jun 2021), 1–1. DOI: https://doi.org/10.1109/TKDE.2021.3090866

[23] Zhengdong Lu, Todd K. Leen, Yonghong Huang, and Deniz Erdogmus. 2008. A reproducing kernel Hilbert space framework for pairwise time series distances. In *Proceedings of the 25th International Conference on Machine Learning (ICML'08) (ACM International Conference Proceeding Series, Vol. 307)*. ACM, 624–631. DOI: https://doi.org/10.1145/1390156.1390235

[24] Carl Edward Rasmussen. 2003. Gaussian processes in machine learning. In *Proceedings of the Advanced Lectures on Machine Learning, ML Summer Schools 2003 (Lecture Notes in Computer Science, Vol. 3176)*. Springer, 63–71. DOI: https://doi.org/10.1007/978-3-540-28650-9_4

[25] Yulia Rubanova, Tian Qi Chen, and David Duvenaud. 2019. Latent ordinary differential equations for irregularly-sampled time series. In *Proceedings of the Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems, (2019) NeurIPS'19*. 5321–5331.

[26] Satya Narayan Shukla and Benjamin M. Marlin. 2019. Interpolation-prediction networks for irregularly sampled time series. In *Proceedings of the 7th International Conference on Learning Representations, (ICLR'19)*. Retrieved from OpenReview.net. https://openreview.net/forum?id=r1efr3C9Ym.

[27] Huan Song, Deepta Rajan, Jayaraman J. Thiagarajan, and Andreas Spanias. 2018. Attend and diagnose: Clinical time series analysis using attention models. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence,*

(AAAI-18), the 30th Innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI'18). AAAI Press, 4091–4098.

[28] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems* (2014). 3104–3112.

[29] Qingxiong Tan, Mang Ye, Baoyao Yang, Siqi Liu, Andy Jinhua Ma, Terry Cheuk-Fung Yip, Grace Lai-Hung Wong, and Pong Chi Yuen. 2020. DATA-GRU: Dual-attention time-aware gated recurrent unit for irregular multivariate time series. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence, AAAI 2020, The 32nd Innovative Applications of Artificial Intelligence Conference, IAAI'20, The 10th AAAI Symposium on Educational Advances in Artificial Intelligence, (EAAI'20)*. AAAI Press, 930–937.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems*, Vol. 30. Curran Associates, Inc.

[31] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *Proceedings of the Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems (NeurIPS'19)*. 5754–5764.

[32] Changchang Yin, Ruoqi Liu, Dongdong Zhang, and Ping Zhang. 2020. Identifying sepsis subphenotypes via time-aware multi-modal auto-encoder. In *Proceedings of the KDD'20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, 862–872. DOI: https://doi.org/10.1145/3394486.3403129

[33] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. 2021. A transformer-based framework for multivariate time series representation learning. In *Proceedings of the KDD'21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, 2114–2124. DOI: https://doi.org/10.1145/3447548.3467401

[34] Xianli Zhang, Buyue Qian, Shilei Cao, Yang Li, Hang Chen, Yefeng Zheng, and Ian Davidson. 2020. INPREM: An interpretable and trustworthy predictive model for healthcare. In *Proceedings of the KDD'20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, 450–460. DOI: https://doi.org/10.1145/3394486.3403087

Received August 2021; revised December 2021; accepted January 2022