A Hybrid Approach for Irregular-Time Series Prediction using Electronic Health Records: an Intensive Care Unit Mortality Case Study

SHAOJIE ZHONG, College of Computing and Data Science, Nanyang Technological University, Singapore
LI RONG WANG, College of Computing and Data Science, Nanyang Technological University Centre for Frontier AI Research, Agency for Science, Technology and Research, Singapore
ZHUOXUAN ZHAN, College of Computing and Data Science, Nanyang Technological University, Singapore
YIH YNG NG, Department of Preventive and Population Medicine, Tan Tock Seng Hospital Lee Kong Chian School of Medicine, Nanyang Technological University, Singapore
XIUYI FAN, Lee Kong Chian School of Medicine College of Computing and Data Science, Nanyang Technological University Centre for Medical Technologies & Innovations, National Health Group, Singapore

In recent years, deep learning has gained traction in medical research, particularly in addressing challenges related to predicting outcomes from irregularly sampled time-series data. This irregularity, stemming from inconsistent follow-up appointments and vital sign recordings, poses significant hurdles for traditional machine learning techniques reliant on regularly sampled data. To overcome this, researchers have developed two main categories of methods: interpolation-based and non-interpolation-based models. Our study proposes STraTS-mTAND, a novel approach that integrates techniques from both categories to predict outcomes from irregularly sampled time-series data. By leveraging the strengths of STraTS, a non-interpolation model, and mTAND, an interpolation model, our approach aims to mitigate their respective limitations. Evaluation on datasets like the PhysioNet Challenge 2012 and MIMIC-III demonstrates superior performance compared to existing methods, showing improvements in both PR-AUC and ROC-AUC metrics. Notably, our approach exhibits robustness with less training data and performs effectively with sparser and more irregular time series – characteristics inherent in recorded medical data. Additionally, our analysis offers insights into optimizing predictive modelling in healthcare, potentially reducing costs associated with data extraction and management, shortening operational cycles for model development and deployment, and enhancing digital agility and sustainability within the healthcare sector.

CCS Concepts: • **Applied computing → Health informatics**.

Additional Key Words and Phrases: Irregular Time-Series, Vital Signs, ICU Mortality

Authors' addresses: Shaojie Zhong, szhong005@e.ntu.edu.sg, College of Computing and Data Science, Nanyang Technological University, Singapore; Li Rong Wang, lirong002@e.ntu.edu.sg, College of Computing and Data Science, Nanyang Technological University and Centre for Frontier AI Research, Agency for Science, Technology and Research, Singapore; Zhuoxuan Zhan, zhuoxuan001@e.ntu.edu.sg, College of Computing and Data Science, Nanyang Technological University, Singapore; Yih Yng Ng, yih_yng_ng@ttsh.com.sg, Department of Preventive and Population Medicine, Tan Tock Seng Hospital and Lee Kong Chian School of Medicine, Nanyang Technological University, Singapore; Xiuyi Fan, xyfan@ntu.edu.sg, Lee Kong Chian School of Medicine and College of Computing and Data Science, Nanyang Technological University and Centre for Medical Technologies & Innovations, National Health Group, Singapore.

ACM 2637-8051/2025/6-ART
https://doi.org/10.1145/3743689

1 INTRODUCTION

Irregular-time series prediction is of paramount importance in healthcare due to the inherent variability and complexity of patient data, particularly within Electronic Health Records (EHR) systems[1]. Unlike regular time series data, which follows a consistent sampling interval, EHR data often exhibits irregular patterns, with measurements recorded at varying intervals based on patient encounters, diagnostic tests, treatments, and other clinical activities [13]. This irregularity poses a significant challenge for traditional time series forecasting methods, as they may struggle to effectively capture the underlying trends and patterns within EHR data [20]. In the context of healthcare, accurate prediction of future events such as disease progression, medication adherence, or patient outcomes is essential for guiding clinical decision-making, optimizing resource allocation, and improving patient outcomes [48]. Therefore, developing robust predictive models capable of handling irregular-time series data extracted from EHRs is critical for unlocking the full potential of data-driven healthcare interventions.

ICU mortality prediction is a critical tool for clinicians to optimize patient care and resource allocation [9]. Most patients admitted to the ICU are facing life-threatening conditions. Treating these conditions is often cognitively complex and demands a high volume of time-sensitive medical decisions, thereby increasing the likelihood of errors. Hence, automated real-time predictions can be extremely beneficial in assisting clinicians in making medical decisions. Accurate prediction models enable healthcare professionals to identify patients at heightened risk of mortality, facilitating timely interventions and personalized treatment plans tailored to individual patient needs. This proactive approach also enhances communication between clinicians, patients, and their families regarding prognosis and treatment decisions [8].

From a computer science standpoint, ICU mortality prediction drives innovation in predictive modeling and machine learning techniques, enabling the development of sophisticated algorithms capable of analyzing irregular data collected from diverse data sources, identify complex patterns, and extract actionable insights to support clinical decision-making in real-time [22, 31]. As illustrated in Figure 1, the data collected in the ICU are characterized with a few distinct properties summarised as follows.

(1) The time intervals between observations lack regularity, a characteristic commonly observed in multivariate irregularly sampled time-series data.
(2) Observation timings show asynchrony across variables, resulting in variations in recorded timestamps for each variable within the multivariate irregularly sampled time-series data.
(3) Within the multivariate irregularly sampled time-series data, there may be instances where a patient's time-series lacks observations for specific variables.

The highly irregular and sparse nature of the observations presents significant challenges when attempting to apply traditional machine learning models for prediction. These models typically rely on time-series data with regular intervals. While one common approach to address this issue involves transforming the irregular time-series into a regular one using imputation techniques, such as forward-filling or mean-imputation [43], this method comes with its own set of drawbacks. Imputations may introduce biases into the data and consequently reduce the accuracy of predictive models [21, 27]. This necessitates the exploration of alternative methods that can effectively handle the unique characteristics of such data while preserving predictive accuracy.

To address the limitations of the aforementioned methods, recent works have focused on the development of machine learning solutions that do not require inputs to be transformed into a regular time-series. These solutions can be categorized into two types: *interpolation-based* and *non-interpolation-based* models. Interpolation involves estimating unknown values based on known data points [28]. An example of an interpolation-based model is the Discretized Multi-Time Attention (mTAND) [38], which includes a learnable interpolation component and aims to learn the temporal similarities.

On the other hand, non-interpolation-based models directly use irregularly sampled time series as inputs [5]. For instance, such approaches take input as a sequence consisting of observed values, missing data indicators, and

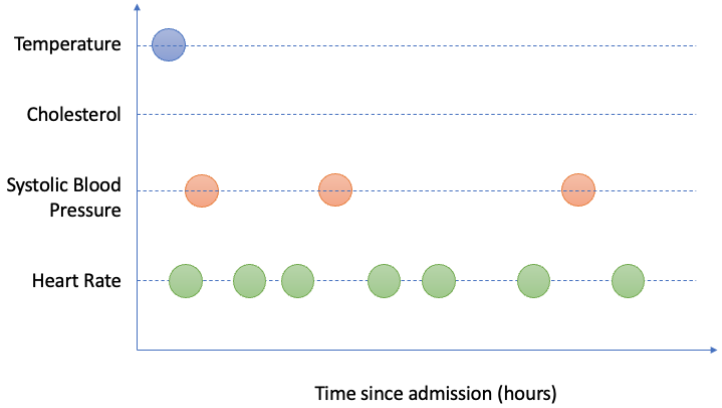

Fig. 1. An illustration of multivariate irregularly sampled time-series in a healthcare setting reveals variability in the time intervals between observations of different features. Moreover, the number of observations for each feature can vary widely, ranging from zero to any positive integer. This variability underscores the complexity and heterogeneity inherent in healthcare data, presenting unique challenges for analysis and prediction tasks.

time intervals. An example of a non-interpolation-based model is the Self-supervised Transformer for Time-Series (STraTS) [41], which encodes each observation individually and does not depend on interpolation. While both methods are different, they both achieve significant success in predicting ICU mortality from irregular data.

In this project, we propose a new solution, STraTS-mTAND, that utilizes both approaches to improve ICU mortality prediction performance. The idea is that interpolation is unlikely to be perfect, whereas non-interpolation misses the opportunity to extract information from correct interpolations. Moreover, since the "correctness" of interpolation can be learned at the same time as the model training, it is worthwhile to develop a hybrid model that contains both an interpolation and non-interpolation component.

The contributions of our work are as follows:

(1) Incorporated interpolation and non-interpolation models into a single model, STraTS-mTAND, to tackle the in-hospital mortality problem.
(2) Evaluated our proposed solution on two EHR datasets, PhysioNet Challenge 2012 [39] and MIMIC-III [24], and compared them to its components and other competitive baseline models for in-hospital mortality prediction.
(3) Investigates the in-hospital mortality predictability problem, emphasizing the significance of adjusting prediction time windows and integrating static information to enhance prediction accuracy.

The remainder of this paper is structured as follows. In Section 2, we provide a review of the existing literature in the field of irregular-time prediction. Section 3 presents the prediction problem and our proposed solution. Our experimental methodology, including the testing of our proposed solution and the evaluation of performance, is detailed in Section 4. Section 5 discusses additional experiments aimed at gaining more insights in the ICU mortality problem. Finally, Section 6 discusses our findings and conclusions, while also suggesting potential avenues for future research.

2 RELATED WORKS

The characteristic of irregularly sampled time-series data poses a problem to machine learning models as these models typically accept data of fixed feature size [46]. Recurrent neural networks (RNNs), which can accept

time-series of different lengths, also struggle with irregularly sampled time-series data as RNNs assume that each sample in the sequence is taken at regular time intervals. Due to these limitations, new methods to utilise irregularly sampled time-series have been researched over the years to better capture the structure of irregularly sampled time-series data.

A simple approach to deal with the problems of irregular sampling is to transform the dataset into a regularly sampled dataset. Lipton et al. [26] and McDermott et al. [29] grouped the time-series into one-hour time intervals and used imputation strategies such as forward-filling and zero-imputation strategies in order to transform the irregular intervals into regular intervals. Other methods to impute the missing values include using Gaussian processes to model the time-series [15] and re-sampling methods to decide which data to impute [12]. While such an approach is simple, we have to handle situations when there are no observations within the time interval or if there is more than one observation. An alternative approach is to construct a model that accepts the irregularly sampled dataset as input. As mentioned in the previous section, these models can be categorised into two types, interpolation-based and non-interpolation-based models.

Interpolation-based models can be described as models that aim to generate embeddings at different timestamps. Che et al. [5] modified the inputs of the Gated Recurrent Unit (GRUs) [11] to include missing data indicators to indicate the lack of an observation and time interval since the last observation in addition to the observed values to allow the GRU to capture the irregularity of the time-series. They also proposed the Gated Recurrent Unit with trainable Decay (GRU-D) [5] that decays the input and hidden states when there are unobserved time intervals. Modifications to Long Short-Term Memory(LSTM) [18] have also been proposed. Pham et al. [33] proposed to modify the forget gate and Neil et al. [32] proposed a new time gate to regulate the access to the hidden and cell states of LSTM. To better capture the complex patterns in irregular time series, Zhou et al. [49] proposed a 2D cross-regression (2DCR) method to exploit the correlation across both temporal and variable domains and, combined with a bi-directional RNN, to jointly perform data imputation and mortality prediction. For the inputs of these RNN models, the sequence length is the number of unique timestamps in the irregular time-series, which can be quite large, hence it may result in long training and inference time. To circumvent this issue, Shukla & Marlin proposed Discretized Multi-Time Attention (mTAND) [38] which uses an attention [42] mechanism instead to generate embeddings. To further explore the challenge, some methods extend the models for processing time series in a continuous manner. Wi et al. [44] redesigned the time series autoencoder based on Neural Controlled Differential Equations and proposed their RNN-based model, called continuous-time autoencoder (CTA), which can be seen as creating infinitely many autoencoders for the time series. Chen et al. [6] adopted the ODE-GRU model for irregular human motion prediction. With a modified fusion connection and loss function, the model performed well in Human3.6 and CMU-Mocap datasets. Chen et al. [7] presented a novel transformer variant for modeling dynamic systems of irregular time series data, combining the modeling abilities of continuous dynamics of Neural Ordinary Differential Equations (NODE). Chauhan et al. [4] also presented an attention-based model to deal with irregular time series in EHR, similarly using NODE to generate regular time series, but with a learnable latent and cross-attention mechanism to reduce the computation cost of the self-attention layers. Furthermore, while most of the interpolation-based methods focus on imputing all the missing values or using all the observations to impute, Blazquez-Garcia et al. [2] argued that the quality of time points matter, and proposed a selective imputation method based on Multi-task Gaussian Process (MGP), which can serve as a preprocessing step for all methods. Additionally, inspired by the current trends in deep learning, the construction of pre-trained models has also emerged for irregular time series tasks. Chowdhury et al. [10] proposed PrimeNet to learn a self-supervised representation for irregular multivariate time-series, designing a time-sensitive contrastive learning and data reconstruction task to pre-train a model.

Non-interpolation-based models can be described as models that do not aim to generate representations at different timestamps instead, they view the problem of classifying the time-series as classifying a set of observations. Horn et al. [19] proposed the Set Functions for Time Series (SeFT) that take in inputs as observation

triplets. It utilises differentiable set functions to summarize each observation triplet. It then passed the summarized observations to an attention mechanism to learn the importance of each observation, which is then used in the final classification layer. Self-supervised Transformer for Time-Series (STraTS) [41] proposed by Tipirneni & Reddy improves on the SeFT by using learnable embeddings. TransEHR, proposed by Xu et al. [45], used a similar architecture of SeFT, but with self-supervised learning. Compared with STraTS, which also incorporates self-supervised learning, TransEHR conducted more pretext tasks for pre-training.

2.1 Discretized Multi-Time Attention (mTAND)

Shukla and Marlin [38] proposed an attention-based mechanism that uses irregular timings where an observation was recorded to interpolate the time-series and estimate the time-series at regular predefined time points. In the attention mechanism, the irregular timings form the key, the regular predefined time points form the query and the value of the observations forms the value. The output of the attention layer is the estimated values of time-series variables at the regular predefined time points. Instead of selecting the imputation techniques to use, the attention mechanism provides learnable parameters to understand the temporal similarity from the training data. In their experiments, their way of interpolation outperforms other imputation methods for interpolating the PhysioNet Challenge 2012 dataset. Through this learnable interpolation, we can obtain a regularly sampled, fixed-size representation of the time-series, solving the problem of irregularity and misalignment of the observations. This fixed-size representation can then be used in RNNs to perform other prediction tasks such as classification.

Their use of attention focuses on generating the embeddings of the time-series at regular timestamps to allow the classifier to learn the trajectory of the patients condition. This is drastically different from STraTS. While this method has proven successful, the interpolation may result in noise and loss of information from the irregularity of the dataset. Hence in our project, we aim to utilise both STraTS and mTAND representation of the time-series to provide the final prediction layer with a more robust representation of the time-series.

2.2 Self-supervised Transformer for Time-Series (STraTS)

STraTS [41] treats the time-series as sets of observational triplets of time, feature and feature value. It considers the time and feature of an observation as the positional information and adds the time and feature embeddings to the observed value embeddings. This creates a vector for each observation that contains information on the time, feature and observed value. The observation embeddings are then passed through Transformer [42] blocks to encode the entire time-series, hence drawing dependencies between observations. Their solution resulted in an improvement of 1.0% and 0.7% in the ROC-AUC score for the MIMIC III and PhysioNet Challenge 2012 in-hospital mortality dataset respectively when compared to using SeFT.

Besides proposing a new architecture, they also utilised self-supervised learning to improve the performance on prediction task. The self-supervised learning task uses a shorter observation window from each time-series and aims to forecast the time-series variables in a short window after the observation window. Their method does not perform any imputation or require additional missing data indicators, instead it encodes each observation individually. This method simplifies the problem by removing the irregularity and sparsity of the time-series from the input data. However, when the observations have a large time-interval, it may be difficult for the model to project the trajectory of the patients condition. Hence having interpolation can assist the model in providing some information when the observations are far apart.

Table 1 summarises the different techniques developed to tackle prediction problems involving irregularly sampled time-series.

Table 1. Summary of related works. "Interpolation based" indicates whether the technique aims to transform time-series into regular intervals. "Underlying Architecture Type" indicates the underlying neural network architecture.

Technique	Interpolation based	Underlying Architecture Type
GP-VAE [15]	Yes	Autoencoders
GRU-D [5]	Yes	RNN
Phased LSTM [32]	Yes	RNN
SeFT [19]	No	Attention
STraTS [41]	No	Attention
mTAND [38]	Yes	Attention

3 PROPOSED SOLUTION

Interpolation-based methods and non-interpolation-based methods have both demonstrated success; however, each approach has its drawbacks. Interpolation-based methods may produce inaccurate interpolations, introducing noise into the data. On the other hand, non-interpolation-based methods rely solely on observed data for predictions, potentially overlooking valuable information recoverable through interpolations. Therefore, our proposed solution combines models from both approaches to address the irregularly sampled time-series prediction problem. By integrating these two models, we leverage their distinct encoding methods, each capturing a unique representation of the time-series data. This fusion enables us to create a more robust representation for improved prediction accuracy. In this chapter, we begin by defining the problem and input parameters, followed by an overview of the architecture of our proposed solution.

In our investigation, we concentrated on two prominent datasets: the PhysioNet Challenge 2012 dataset [39] and the MIMIC-III dataset [24]. Each instance within these datasets corresponds to an ICU stay and encompasses a spectrum of demographic information, including age and gender, alongside medical data such as heart rate and temperature. Further elaboration on these datasets is provided in Section 4.1.

3.1 Problem Definition

Considering a supervised learning task, where the dataset $\mathcal{D}$ contains N instances. This dataset has V time-series variables and D static variables. Time-series variables refer to variables that change with time. In the MIMIC-III dataset, examples of time-series variables are heart rate or temperature. Static variables refer to variables that stay approximately constant through the time-series. Examples of static variables in the MIMIC-III dataset are age and gender.

The dataset $\mathcal{D} = \{(\mathbf{T}_j^s, \mathbf{T}_j^m, \mathbf{d}_j, y_j) | j = 1, \ldots, N\}$ consists of four main components. The j^{th} instance in the dataset consists of the target variable y_j, static variables vector $\mathbf{d}_j \in \mathbb{R}^D$ and two representations of the time-series variables. We elaborate on the two representations in the following paragraphs. Consider an instance in the dataset where the time-series has n^s observations recording in n^t unique timestamp.

The first time-series representation is $\mathbf{T}^m = (\mathbf{t}^m, \mathbf{X}^m, \mathbf{M}^m)$ as shown in Figure 2. $\mathbf{t}^m \in \mathbb{R}_{\geq 0}^{n^t}$ is the observation time vector that lists the timestamps where an observation is recorded. $\mathbf{X}^m \in \mathbb{R}^{n^t \times V}$ is the observation time matrix which stores the value of the observations. $\mathbf{M}^m \in \{0, 1\}^{n^t \times V}$ is the observation time mask that indicates 1 when an observation in $\mathbf{X}^m$ is observed and 0 when unobserved. The observation for the k^{th} variable recorded at time t_g^m, where $g \in \{1, 2, ..., n^t\}$ can be found at $\mathbf{X}_{g,k}^m$ with $\mathbf{M}_{g,k}^m = 1$. For variables that are not recorded at time t_g^m, $\mathbf{X}_{g,k}^m = 0$ with $\mathbf{M}_{g,k}^m = 0$. The second representation of the time-series represents the time-series as sets of

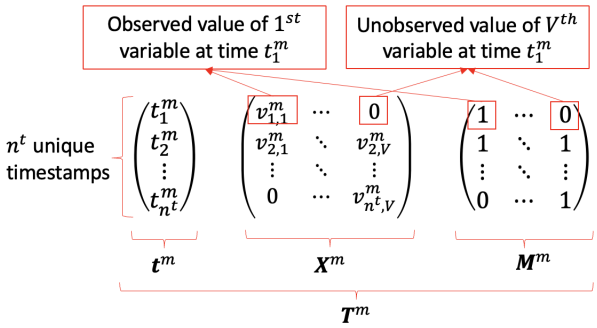

Fig. 2. An example of the $\mathbf{T}^m$ time-series.

observational triplets $\mathbf{T}^s = \{(t_k^s, f_k^s, v_k^s)\}_{k=1}^{n^s}$ as shown in Figure 3. Each set of observation consists of the time $t_k^s \in \mathbb{R}_{\geq 0}$, feature $f_k^s \in \mathcal{F}$ and the feature value $v_k^s \in \mathbb{R}$.

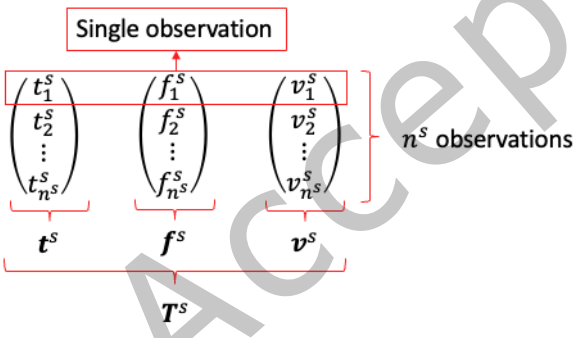

Fig. 3. An example of the $\mathbf{T}^s$ time-series.

In the first representation $\mathbf{T}^m$, the time-series is represented in a value matrix $\mathbf{X}^m$, where each column represents the observed value of a variable. Due to the irregular nature of the time-series, it results in this matrix being sparse and this sparsity varies throughout the dataset. Whereas in the second representation $\mathbf{T}^s$, the time-series is represented by observations. This removes the sparsity of the time-series, resulting in a compact representation. However, due to the fixed-size observation value matrix representation $\mathbf{X}^m$ in $\mathbf{T}^m$, we do not need to perform any encoding to represent the variables as it is already represented by the column in the matrix, whereas in the second representation, we need an additional vector $\mathbf{f}^s$ to represent the variables.

3.2 Proposed Model

Our proposed model(STraTS-mTAND) modifies the STraTS, a non-interpolation-based model, by incorporating mTAND, an interpolation-based model. In this section, we refer to the STraTS module as NI-module (Non-Interpolation module) and mTAND module as the I-module (Interpolation module). Figure 4 shows the overview architecture of STraTS-mTAND. In this section, we will be elaborating on the individual components. It should be noted that, for the NI-module, we adopt the original STraTS model directly, and for the I-module, we modified the Time Embedding in the original mTAND model with the CVE module used in STraTS.

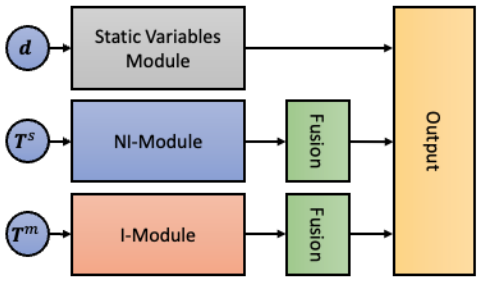

Fig. 4. Overview of the STraTS-mTAND partitioned into 6 modules, Static Variable Module, NI-Module, I-Module, 2 Fusion module and an Output module.

3.2.1 NI-module Embeddings. The NI-module takes the time-series as sets of observational triplets $\mathbf{T}^s = \{(t_k^s, f_k^s, v_k^s)\}_{k=1}^{n^s}$ Each set of observation consists of the time t_k^s, feature f_k^s and the feature value v_k^s. The output of the NI-module is the observations embedding $\mathbf{C} \in \mathbb{R}^{n^s \times d}$. Figure 5 shows the overview of the process used to generate the observations embeddings $\mathbf{C}$.

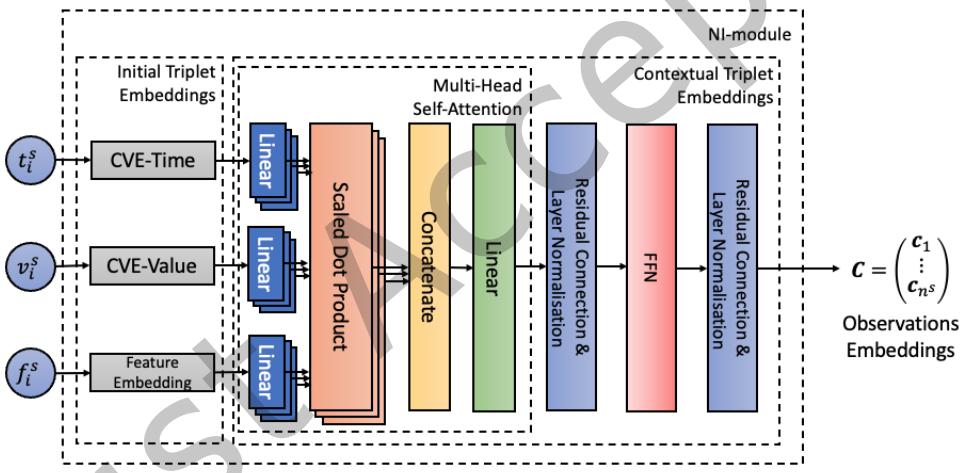

Fig. 5. Breakdown of the NI-Module. The NI-Module consist of 3 parts, Initial Time Embeddings, Multi-Head Self Attention and Contextual Triplet Embeddings. This module outputs the observations embeddings.

Initial Triplet Embedding. Each component of the k^{th} observation is first encoded individually. As the feature f_k is categorical, it is encoded using a lookup table that converts it into feature embedding $\mathbf{e}_k^f \in \mathbb{R}^d$. Time t_k^s and feature value v_k^s are continuous, hence they are encoded using the Continuous Value Embedding(CVE) module. The CVE module consists of two dense layers with the hidden layer having a dimension of $\sqrt{d}$ and an output dimension of d. The parameters of the CVE modules are $\mathbf{W}_1^{CVE} \in \mathbb{R}^{\sqrt{d}}$, $\mathbf{b}_1^{CVE} \in \mathbb{R}^{\sqrt{d}}$ and $\mathbf{W}_2^{CVE} \in \mathbb{R}^{d \times \sqrt{d}}$.

$$CVE(x) = \mathbf{W}_2^{CVE} \tanh(\mathbf{W}_1^{CVE} x + \mathbf{b}_1^{CVE}) \tag{1}$$

Time and feature values each have their own *CVE* module that encodes them into time embedding $\mathbf{e}_k^t \in \mathbb{R}^d$ and value embedding $\mathbf{e}_k^v \in \mathbb{R}^d$. The time embedding $\mathbf{e}_k^t$, feature embedding $\mathbf{e}_k^f$ and feature value embedding $\mathbf{e}_k^v$ are added together to form the initial triplet embedding $\mathbf{i}_k \in \mathbb{R}^d$ that represents the time t_k^s, feature f_k^s and feature value v_k^s of the k^{th} observation.

$$\mathbf{i}_k = \mathbf{e}_k^t + \mathbf{e}_k^f + \mathbf{e}_k^v \tag{2}$$

This is done for all n^s observations in the time-series, resulting in an initial triplet embeddings $\mathbf{I} = \{\mathbf{i}_1, \ldots, \mathbf{i}_{n^s}\} \in \mathbb{R}^{n^s \times d}$ for the entire time-series. What we have obtained here are embeddings for each observation that are encoded individually. After obtaining the initial triplet embedding by adding the time embedding, value embedding and feature embedding, the initial triplet embedding will be passed to the Multi-Head Self-Attention layer, and the output of h heads will be concatenated and sent to a linear layer.

Contextual Triplet Embedding. The initial triplet embeddings $\mathbf{I}$ are then passed to the Contextual Triplet Embedding module, which consists of M Transformer [42] blocks. It outputs the contextual triplet embeddings $\mathbf{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_{n^s}\} \in \mathbb{R}^{n^s \times d}$. Each block consists of a Multi-Head Self-Attention layer with h heads followed by two dense layers with the hidden layer having a dimension of $2d$. The calculations for each block are as follows.

The Multi-Head Self-Attention layer takes in the input embeddings $\mathbf{I} \in \mathbb{R}^{n^s \times d}$ and outputs self-attention embeddings $\mathbf{H} \in \mathbb{R}^{n^s \times d}$ between the observations.

$$\mathbf{H}_j = \text{softmax}\left(\frac{(\mathbf{I}\mathbf{W}_j^q)(\mathbf{I}\mathbf{W}_j^k)^T}{\sqrt{d/h}}\right)(\mathbf{I}\mathbf{W}_j^v) \quad \text{for } j = 1, \ldots, h \tag{3}$$

$$\mathbf{H} = (\mathbf{H}_1 \circ \cdots \circ \mathbf{H}_h)\mathbf{W}_c \tag{4}$$

The Multi-Head Self-Attention layer projects the input embeddings into h different queries, keys and values using $\{\mathbf{W}_j^q, \mathbf{W}_j^k, \mathbf{W}_j^v\} \in \mathbb{R}^{d \times d_h}$ respectively for each head$_j$. The queries and keys are used to compute the attention weight matrix. Drop Attention is applied to the attention weight matrix to randomly drop elements during training to reduce overfitting [47]. The attention weights are then used to compute the weighted averages of the embedings between n^s observations. Lastly, the outputs of each head H_j are concatenated and reduced to the original dimension d using $\mathbf{W}^c \in \mathbb{R}^{(hd_h) \times d}$, resulting in $\mathbf{H}$. Residual connection and layer normalisation are then performed on $\mathbf{H}$ before it is passed into a two-layer feed-forward network(FFN).

$$FFN(\mathbf{X}) = \mathbf{W}_2^{FFN}\text{ReLU}(\mathbf{W}_1^{FFN}\mathbf{X} + \mathbf{b}_1^{FFN}) + \mathbf{b}_2^{FFN} \tag{5}$$

Dropout, residual connection and layer normalisation are applied to the output of FFN.

The above process is repeated through the M Transformer blocks with the inputs for subsequent Transformer blocks as the output of the previous, resulting in the observations embeddings $\mathbf{C}$.

In this process, the self-attention mechanism draws dependencies between the observations.

3.2.2 I-module Embeddings. For the I-module, it takes the irregularly sampled time-series as $\mathbf{T}^m = (\mathbf{t}^m, \mathbf{X}^m, \mathbf{M}^m)$. The inputs are also accompanied by a regular interval time query $\mathbf{t}^q \in \mathbb{R}^{n^q}$. The output of the I-module is the regular time embeddings $\mathbf{R} \in \mathbb{R}^{n^q \times d}$. Figure 6 shows the overview of the process used to generate the regular time embeddings $\mathbf{R}$.

Regular Interval Time Query. The regular interval time query $\mathbf{t}^q$ is a vector with consecutive values being separated by a constant difference. $\mathbf{t}^q$ depends on the length of the time-series problem t and the number of intervals n^q which is a hyperparameter.

The length of the time-series problem t is specific to the prediction problem and refers to the time at which all time-series end. In the case of the PhysioNet Challenge 2012, the in-hospital mortality task uses the first 48 hours of data in its prediction, hence the length of the time-series problem is 48 hours.

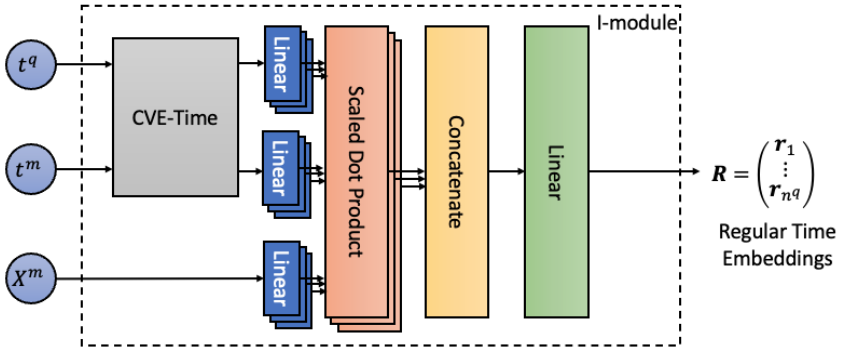

Fig. 6. Breakdown of the I-Module. The I-Module consist of 2 parts, Initial Time Embeddings and Multi-Head Self Attention. It outputs the regular time embeddings.

The number of intervals n^q is a hyperparameter and it refers to the number of time points that we want the model to interpolate using the mTAND module.

With the two parameters, the time query is defined as the following:

$$\mathbf{t}^q = \left(0, \dots \frac{i \cdot t}{n^q}, \dots, t\right)^T \text{ for } i \in \{1, 2, 3, ..., n^q\} \tag{6}$$

This time query will inform the attention mechanism in the mTAND module at which time point to produce interpolation.

Initial Time Embedding. The time inputs, $\mathbf{t}^m$ and $\mathbf{t}^q$, are first encoded using the *CVE* module. This *CVE* module uses the same architecture of NI-module (Equation 1) but will learn its own parameters. This *CVE* module encodes $\mathbf{t}^m$ into the time key embeddings $\mathbf{E}^{t^m} \in \mathbb{R}^{n^m \times d}$ and t^q into the time query embeddings $\mathbf{E}^{t^q} \in \mathbb{R}^{n^q \times d}$.

Regular Time Embedding. After encoding the time into vectors, the inputs are passed to a Multi-Head Attention mechanism with h heads followed by a series of dense layers. It takes in three matrices as inputs, $\mathbf{E}^{t^q}$, $\mathbf{E}^{t^m}$ and $\mathbf{X}^m$, and outputs a fixed-size representation of the time series $\mathbf{R} \in \mathbb{R}^{n^q \times d}$ at regular time points defined by the time query $\mathbf{t}^q$. The calculations are as follows.

The Multi-Head Attention Layer takes in the time query embeddings $\mathbf{E}^{t^q}$ as the query, the time embeddings of the observations $\mathbf{E}^{t^m}$ as the key and the time matrix $\mathbf{X}^m$ as the value for the attention.

$$\mathbf{H}_j = \text{softmax}\left(\frac{(\mathbf{t}^q\mathbf{W}_j^q)(\mathbf{t}^m\mathbf{W}_j^k)^T}{\sqrt{d/h}}\right)(\mathbf{X}^m\mathbf{W}_j^v) \text{ for } j = 1, \dots, h \tag{7}$$

$$\mathbf{H} = (\mathbf{H}_1 \circ \cdots \circ \mathbf{H}_h) \tag{8}$$

Similar to the Multi-Head Self-Attention layer, the query, key and value get projected into h different subspace using $\mathbf{W}_j^q \in \mathbb{R}^{d \times d_h}, \mathbf{W}_j^k \in \mathbb{R}^{d \times d_h}, \mathbf{W}_j^v \in \mathbb{R}^{V \times V}$ respectively for each head j. Each head j will give us a different representation of the same time series $\mathbf{H}_j \in \mathbb{R}^{n^q \times V}$ and concatenating them along the last axis results in a three-dimension matrix $\mathbf{H} \in \mathbb{R}^{n^q \times V \times h}$. This three-dimensional matrix is then passed through a series of dense layers. The parameters for the series of dense layers are $\mathbf{W}_1^m \in \mathbb{R}^{h \times d}, \mathbf{b}_1^m \in \mathbb{R}^d, \mathbf{W}_2^m \in \mathbb{R}^{\sqrt{V} \times V}, \mathbf{b}_2^m \in \mathbb{R}^{\sqrt{V}}, \mathbf{W}_3^m \in \mathbb{R}^{1 \times \sqrt{V}}$ and $\mathbf{b}_3^m \in \mathbb{R}^d$

$$\mathbf{r}_k^T = \mathbf{W}_3^m(\tanh(\mathbf{W}_2^m \tanh(\mathbf{H}_{k,*,*}\mathbf{W}_1^m + \mathbf{b}_1^m) + \mathbf{b}_2^m)) + \mathbf{b}_3^m \tag{9}$$

$$\mathbf{R} = (\mathbf{r}_1 \circ \cdots \circ \mathbf{r}_{t^q}) \tag{10}$$

In the dense layers, we fuse the different representations $\mathbf{H}_{k,*,*}$ at time $\mathbf{t}_k^q$ to obtain a fixed dimension embedding $\mathbf{r}_k \in \mathbb{R}^d$ for the specified time $\mathbf{t}_k^q$. This is done for all time points and concatenated to obtain the regular time embeddings $\mathbf{R}$.

In this process, the I-module utilises the attention mechanism to produce h representations of the interpolated time-series, which is subsequently used to generate the regular time embeddings $\mathbf{R}$ through a series of dense layers.

3.2.3 Fusion Self-Attention. The observations embeddings $\mathbf{C}$ and regular time embeddings $\mathbf{R}$ each get passed into two different Fusion Self-Attention layers (*FSA*). The Fusion Self-Attention layer is used to generate the attention weights α for the embeddings to fuse them into a vector that represents the information needed in the prediction problem. It consists of two dense layers with parameters $\mathbf{W}_1^{FSA} \in \mathbb{R}^{d\times 2d}$, $\mathbf{b}_1^{FSA} \in \mathbb{R}^{2d}$, $\mathbf{W}_2^{FSA} \in \mathbb{R}^{2d\times 1}$ and performs the following calculations.

$$FSA(\mathbf{E}) = \text{softmax}\left(\tanh\left(\mathbf{E}\mathbf{W}_1^{FSA} + \mathbf{b}_1^{FSA}\right)\mathbf{W}_2^{FSA}\right) \tag{11}$$

For the observations embeddings, the output is $FSA(\mathbf{C}) = \boldsymbol{\alpha}^s \in \mathbb{R}^{n^s}$. For the regular time embeddings, the output is $FSA(\mathbf{R}) = \boldsymbol{\alpha}^m \in \mathbb{R}^{n^q}$. The embeddings are then multiplied with the respective attention weights α to generate the NI-embeddings $\mathbf{e}^s \in \mathbb{R}^d$ and I-embeddings $\mathbf{e}^m \in \mathbb{R}^d$.

$$\mathbf{e}^s = \mathbf{C}^T\boldsymbol{\alpha}^s \tag{12}$$

$$\mathbf{e}^m = \mathbf{R}^T\boldsymbol{\alpha}^m \tag{13}$$

3.2.4 Static Variable Embeddings. Since the static variables $\mathbf{d} \in \mathbb{R}^D$ do not change with time, it is encoded separately from the time-series variables through two dense layers with parameters $\mathbf{W}_1^D \in \mathbb{R}^{1\times 2d}$, $\mathbf{b}_1^D \in \mathbb{R}^{2d}$, $\mathbf{W}_2^D \in \mathbb{R}^{2d\times d}$, $\mathbf{b}_2^D \in \mathbb{R}^d$. This provides us with the static variable embeddings $\mathbf{e}^D \in \mathbb{R}^d$.

$$\mathbf{e}^D = \tanh\left(\tanh\left(\mathbf{d}\mathbf{W}_1^D + \mathbf{b}_1^D\right)\mathbf{W}_2^D\right) + \mathbf{b}_2^D \tag{14}$$

3.2.5 Output Layer. The two sets of time embeddings, $\mathbf{e}^m$ and $\mathbf{e}^s$, and the static variable embeddings $\mathbf{e}^D$ are then concatenated and fed through two dense layers with weights $\mathbf{W}_1^O \in \mathbb{R}^{3d\times V}$, $b_1^O \in \mathbb{R}^V$ and $\mathbf{W}_2^O \in \mathbb{R}^{V\times 1}$, $b_2^O \in \mathbb{R}$, followed by a sigmoid activation function to produce the final output $\hat{y}$.

$$\hat{y} = \text{sigmoid}\left(\left([\mathbf{e}^m \circ \mathbf{e}^s \circ \mathbf{e}^D]\mathbf{W}_1^O + b_1^O\right)\mathbf{W}_2^O + b_2^O\right) \tag{15}$$

All in all, our proposed solution STraTS-mTAND utilised both interpolation-based mTAND and non-interpolation-based STraTS to generate two different representations of the time-series, $\mathbf{e}^s$ and $\mathbf{e}^m$ and use them in the prediction task. Table 2 provides a summary of the mathematical notations mentioned in this section.

4 EXPERIMENTS

We conducted several experiments and evaluated our proposed model against other models using the PhysioNet Challenge 2012 and MIMIC-III dataset for the ICU mortality prediction task. In this section, we present the dataset details, implementation details, and the results of the experiments we performed.

4.1 Datasets and Baseline Models

4.1.1 PhysioNet Challenge 2012. The PhysioNet Challenge 2012 dataset was curated from the MIMIC-II Clinical Database, version 2.6 [36]. The MIMIC-II dataset consists of 25,328 ICU stays which occurred at Beth Israel Deaconess Medical Center between 2001 to 2007. The organisers of the PhysioNet Challenge 2012 curated the

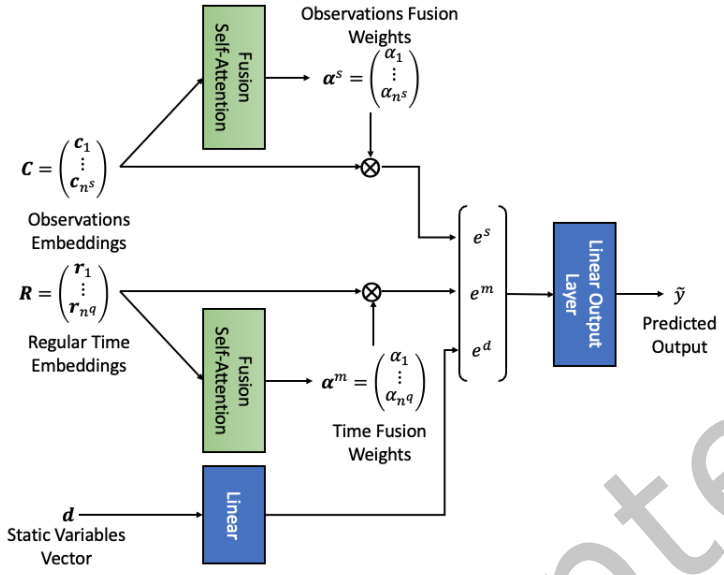

Fig. 7. Breakdown of the Fusion and Output modules in STraTS-mTAND. The observation embeddings and regular time embeddings are fused using the Fusion Self-Attention before concatenating with the static variable embeddings and passing it to the final output layer.

dataset from the MIMIC-II dataset using the following steps. They filtered for ICU stays that last for at least 48 hours with patients aged 16 years old and above on admission and randomly selected 12,000 of such stays.[39] There are a total of 37 time-series variables and 4 static variables. The dataset has multiple prediction problems, such as in-hospital mortality and length of stay. For our project, we will be focusing on the in-hospital mortality prediction. Following the original problem statement in the PhysioNet Challenge, the prediction problem is to predict in-hospital mortality using the first 48 hours of data. The PhysioNet Challenge 2012 data is split into 3 different sets, A, B and C. Following the STraTS paper [41], sets B and C form the training and validation dataset in an 80:20 split and set A forms the test dataset. Table 3 shows the general descriptions of the dataset.

The four static variables serve as general descriptors of the patient and were recorded upon admission. These variables include age, gender, height, and the type of ICU the patient was admitted to. The distribution of these static variables is depicted in Figure 8. There are 37 time-series variables in this dataset and they can be collected at any given time. Each observation has a timestamp that records the timing of the observation since the patient admission into the ICU. Table 15 (appendix) shows the characteristics of the time-series variables for all ICU stays. We can see that there are variables such as Cholesterol appearing only in 7.91% of all stays and White Blood Cell Count appearing on average 3.25 times in 48 hours. This highlights the sparsity and irregularity of the data.

From Figure 9, the distribution of the number of total observations shows that it is slightly right-skewed, with the top 10% ranging from 596 to 1497 observations. 97% of patients have at least 1 observation within the first hour of admission. This percentage gradually decreases with time to 91.7% in the 23rd hour. From Figure 10, the number of hourly observations has generally decreased with time. On further analysis of the trend of hourly observations per patient across time, the correlation plot shows that there is close to no correlation across time. This points out the irregularity of the number of observations across time in the dataset.

Table 2. Mathematical Notations

Notation	Definition
$N \in \mathbb{N}$	# of time-series in dataset
$V \in \mathbb{N}$	# of time-series variables
$D \in \mathbb{N}$	# of static variables
$\mathcal{F}$	Set of V time-series variables
$\mathbf{d} \in \mathbb{R}^D$	Vector for D static variable
$n^s \in \mathbb{N}$	# of observations in the time-series
$t_k^s \in \mathbb{R}_{\leq 0}$	Time of k^{th} observation
$f_k^s \in \mathcal{F}$	Feature of k^{th} observation
$v_k^s \in \mathbb{R}$	Value of k^{th} observation
$\mathbf{T}^s = \{(t_k^s, f_k^s, v_k^s)\}_{k=1}^{n^s}$	Time-series representation for NI-module
$\mathbf{I} = (\mathbf{i}_1, \ldots, \mathbf{i}_{n^s})^T$	Initial Triplet Embeddings for the time-series
$\mathbf{C} = (\mathbf{c}_1, \ldots, \mathbf{c}_{n^s})^T$	Observations Embeddings
$n^m \in \mathbb{N}$	# of timestamps with at least 1 observation
$t^m \in \mathbb{R}^{n^m}$	Observation Time Vector
$\mathbf{X}^m \in \mathbb{R}^{n^m \times V}$	Observation Time Matrix that records values of observations
$\mathbf{M}^m \in \mathbb{R}^{n^m \times V}$	Observation Time Mask that indicates 1 when a variable is observed and 0 otherwise
$\mathbf{T}^m = \{\mathbf{t}^m, \mathbf{X}^m, \mathbf{M}^m\}$	Time-series representation for I-module
$n^q \in \mathbb{N}$	Length of the time query (Hyperparameter)
$\mathbf{t}^q \in \mathbb{R}^{n^q}$	Time query for I-module
$\mathbf{R} = (\mathbf{r}_1, \ldots, \mathbf{r}_{n^q})^T$	Regular Time Embeddings
$\mathbf{e}^s \in \mathbb{R}^d$	NI-Embeddings
$\mathbf{e}^m \in \mathbb{R}^d$	I-Embeddings
$\mathbf{e}^D \in \mathbb{R}^d$	Static Variable Embeddings
$\hat{y} \in [0, 1]$	Output for for binary classification task

Table 3. General characteristic of PhysioNet Challenge 2012 dataset.

	Training	Validation	Test	All
# of ICU Stays	6,392	1,599	3,997	11,988
Avg. span of time-series(hours)	47.3	47.4	47.2	47.3
Avg. # observations/stay	435.7	437.5	434.7	435.6
Max # observations/stay	1191	1156	1497	1497
% of positive labels	14.1	15.6	13.9	14.2

4.1.2 MIMIC-III. The MIMIC-III dataset contains medical records of 46,476 patients admitted to the ICU of Beth Israel Deaconess Medical Center between 2001 and 2012 [17]. To ensure the reproducibility of the dataset, we utilised the MIMIC-III benchmark dataset that was created by Harutyunyan et al. [17]. In their implementation, they selected 17 time-series variables that represent a subset of the 37 variables from the PhysioNet Challenge 2012. Using the same logic, since only age and gender can be found in the MIMIC-III dataset, only 2 static variables are used. They also removed patients below the age of 18 as there is a difference between adult and

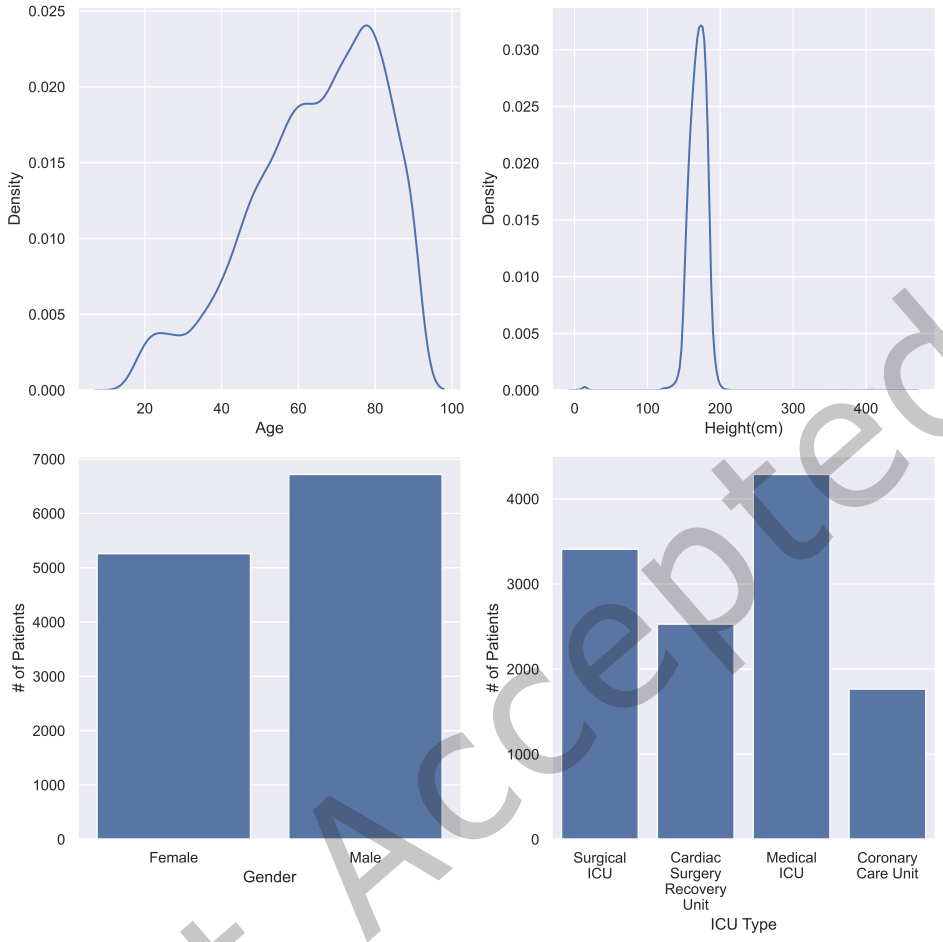

Fig. 8. Distribution of Static Variables. This figure shows a chart for each of the static variables, age, height, gender and ICU type. The age distribution is slightly left-skewed, with the peak at about age 77. The height follows a normal distribution, with the peak at about 160cm. The number of males represents about 55% of the dataset. The medical ICU has the highest number of stays, followed by surgical ICU, cardiac surgery recovery ICU and coronary care ICU.

pediatric physiology [17]. Following the implementation in the STraTS paper, we define the prediction problem as predicting the in-hospital mortality for patients with ICU stays of at least one day. For the splitting of the training, validation and test sets, we utilised Harutyunyan et al. [17] predefined test set. The remaining ICU stays forms the training and validation sets in an 80:20 split. Table 4 shows the general description of the dataset.

Figure 11 shows the distribution of the static variables. The age and gender distribution are relatively similar in both datasets. Similar to the PhysioNet Challenge 2012 dataset, each observation has a timestamp that records the timing of the observation since the patient admission into the ICU. Table 16 (appendix) shows the characteristics of the 17 time-series variables. From Figure 12, the boxplot of the number of observations in each stay shows that the distribution is extremely right-skewed, with the top 1% ranging from 462 to 9533 observations. About 95% of

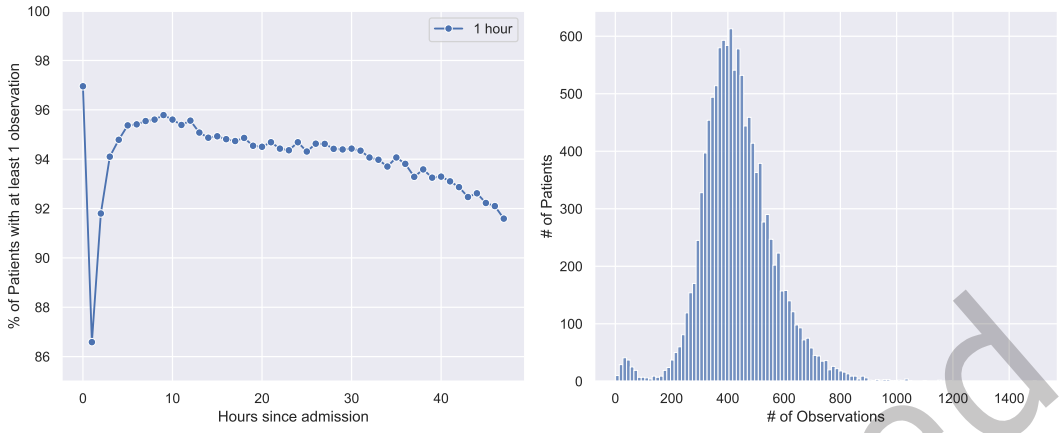

Fig. 9. Distribution of Observations. The line chart (left) illustrates a declining trend in the percentage of patients with at least one observation within an hour over time. Meanwhile, the distribution chart (right) indicates a pattern resembling a normal distribution for the total number of observations, with a median value of 435.

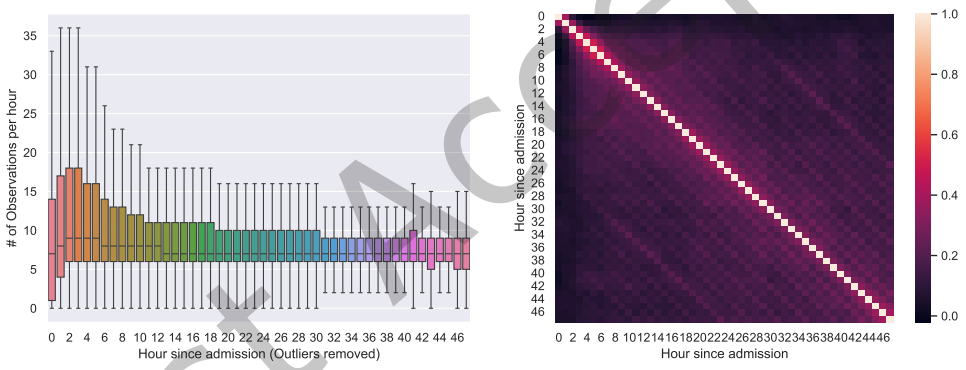

Fig. 10. Number of Observations. The boxplot chart (left) displays the distribution of the number of hourly observations over time, indicating a general decrease in the number of observations. Conversely, the correlation plot (right) reveals minimal correlation with the hourly observations, suggesting limited association between the variables.

the patients have at least one observation per hour most of the time. This is higher as compared to the PhysioNet Challenge 2012 dataset. From Figure 13, the distribution of the number of hourly observations seems constant throughout time. On further analysis of the trend of hourly observation per patient, we plotted a correlation plot of the number of observations between different hours since admission, without ICU stays with more than 462 observations. Similar to the previous dataset, there is little to no correlation across time, thus showing the irregularity of the number of observations across time in the dataset.

4.1.3 Baseline Models. We conducted a performance comparison of STraTS-mTAND against several deep-learning models outlined in Section 2. Notably, mTAND and STraTS are considered state-of-the-art models, as

Table 4. General characteristic of MIMIC-III dataset

	Training	Validation	Test	All
# of ICU Stays	24490	5356	5282	35128
Avg. span of time-series(hours)	23.4	23.4	23.4	23.4
Avg. # observations/stay	234	233	235	234
Max # observations/stay	9365	8859	9533	9533
% of positive labels	10.4	10.9	9.50	10.4

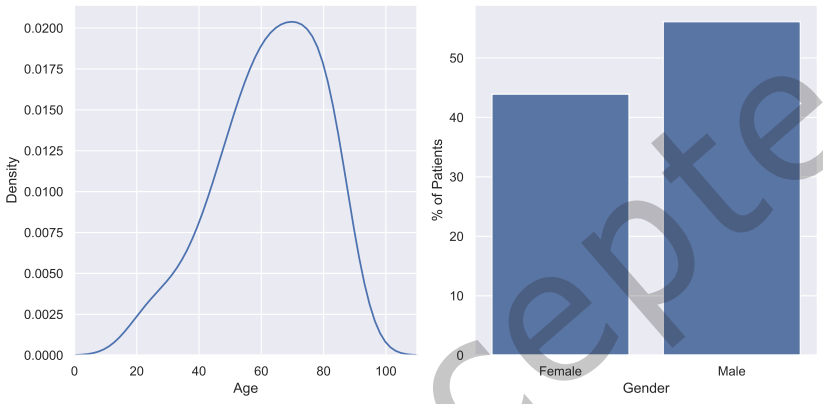

Fig. 11. Distribution of Static Variables. This figure shows a chart for each of the static variables, age and gender. The age distribution is slightly left-skewed, with the peak at age 70. The number of males represents about 55% of the dataset.

demonstrated by previous studies [41**?**], which have shown their superiority over many other models in similar contexts.

(1) GRU with trainable Decay (GRU-D) [5]
(2) Set Functions for Time Series (SeFT) [19]
(3) Discretized Multi-Time Attention (mTAND) [38]
(4) Self-supervised Transformer for Time Series (STraTS) [41]

The time-series variables and static variables for both datasets are normalized to have zero mean and unit variance. All the models are implemented using Keras with TensorFlow backend. For STraTS, we set the maximum number of observations using the 99^{th} percentile. This is done to avoid any memory overflow during batch gradient descent. For mTAND, we adapted the PyTorch implementation from Shukla & Marlin (GitHub) and implemented our version using Keras with Tensorflow backend. Following the implementation of the paper, the observation timestamps rounded to the nearest minute before constructing the time-series matrix. We used the implementation of GRU-D and SeFT from Horn et al. (GitHub) and STraTS from Tipirneni & Reddy (GitHub).

All models were trained with a batch size of 32 using the Adam optimizer [25]. To combat the imbalance class problem, we used a weighted binary cross entropy loss function to provide higher weightage to the positive samples [35]. The training stops when the sum of ROC-AUC and PR-AUC on the validation data does not improve for 10 epochs and the model was restored using the weights with the best validation sum of ROC-AUC and

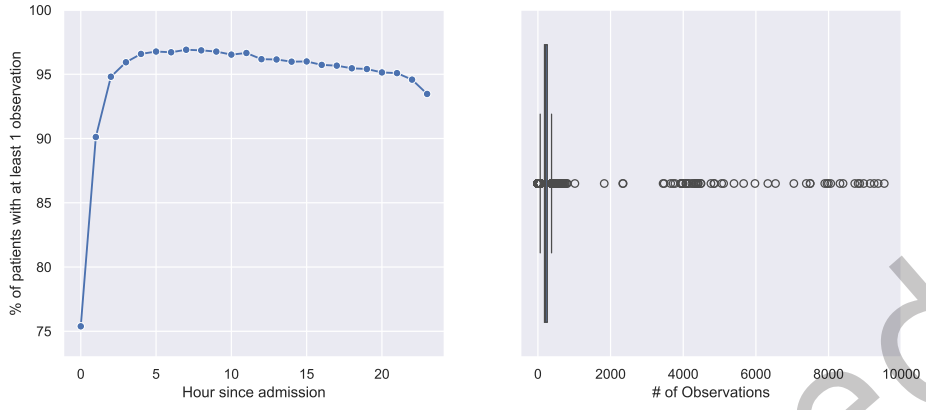

Fig. 12. Distribution of Observations. The line chart (left) shows that the percentage of patients with at least one observation within an hour increases for the first 5 hours and gradually decreases over time. The boxplot chart (right) shows that the number of total observations is extremely right-skewed.

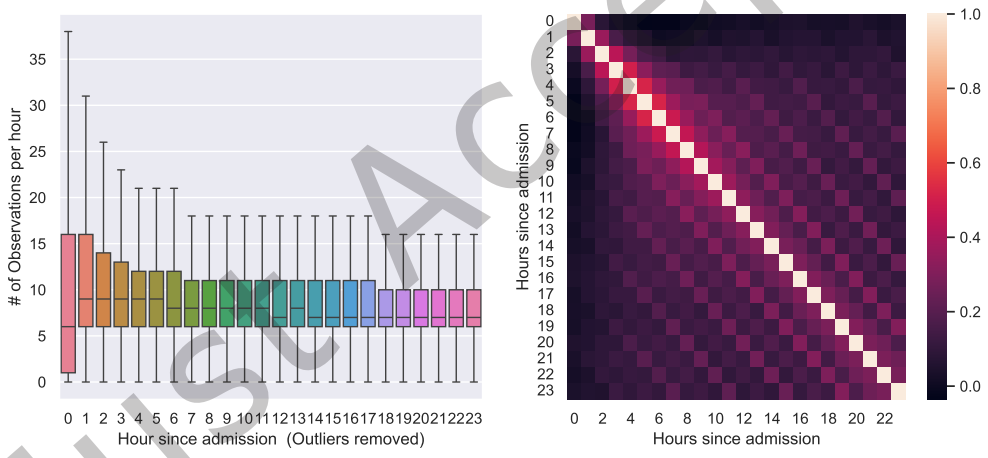

Fig. 13. Number of Observations. The boxplot chart (left) shows the boxplot of the number of hourly observations across time. It shows that the number of observations generally decreases. The correlation plot (right) shows that there is very little correlation with the hourly observations.

PR-AUC before predicting on the test dataset. The hyperparameters used in the experiments are listed in Table 14 (appendix).

4.2 Evaluation Metrics

As the datasets are imbalanced, we used ROC-AUC and PR-AUC to evaluate the performance of the models.

ROC-AUC refers to the area under the Receiver Operating Characteristic (ROC) curve. This curve is constructed by considering all classification thresholds and plotting the true positive rate against the false positive rate at all thresholds. This metric measures the ability of the model to differentiate between the two classes and serves as a combined measure of sensitivity and specificity [16]. In the context of predicting in-hospital mortality, it is essential to maximize sensitivity to capture all potential positive cases, ensuring appropriate allocation of medical resources to critical patients. However, it is equally crucial to maintain specificity to prevent over-prediction of positive cases, which could lead to inefficient resource allocation. Hence a balance between both is required. The higher the ROC-AUC, the better the performance of the model.

$$\text{ROC-AUC} = \int_{x=0}^{x=1} \text{TPR}(\text{FPR}(x))d\text{FPR}, \quad \text{where } x \text{ is the classification threshold} \tag{16}$$

PR-AUC refers to the area under the Precision-Recall curve. Similar to the ROC-AUC, this metric takes into account all classification thresholds and plots the precision against the recall at all thresholds. As this metric uses precision and recall in the underlying calculations, it focuses on the performance of the model on the minority class [14, 37]. Thus this metric is suitable for this prediction problem where predicting the minority class, in-hospital mortality, is more important. The higher the PR-AUC, the better the performance of the model.

$$\text{PR-AUC} = \int_{x=0}^{x=1} \text{Precision}(\text{Recall}(x))d\text{Recall}, \quad \text{where } x \text{ is the classification threshold} \tag{17}$$

4.3 Experiments

We show the performance of our model under various testing conditions.

4.3.1 Overall Comparison between Models. We trained the models using all training data for 10 runs and obtained the following results. The results are shown in Table 5 with **best** performance in **bold**. For the Physionet Challenge 2012 dataset, the STraTS-mTAND outperformed all the baseline models by at least 1% on both the ROC-AUC and PR-AUC metrics. For the MIMIC-III dataset, the average performance on the GRU-D and SeFT are comparable to the STraTS-mTAND, with the ROC-AUC score differing by 0.1% and the PR-AUC score differing by 0.5%. However, the STraTS-mTAND outperformed STraTS in ROC-AUC marginally and 1.5% in PR-AUC. This shows that the inclusion of the mTAND module into the STraTS model provides additional information for the final classification layer. A hypothesis on why the GRU-D and SeFT performed equivalently or better than STraTS-mTAND on the MIMIC-III as compared to the PhysioNet dataset could be because of the reduced number of time-series variables, from 34 to 17.

4.3.2 Generalisation Ability of Models. We tested the generalisation ability of our proposed model to ensure that our additional mTAND module did not result in a worse generalisation of the STraTS model as shown by Tipirneni & Reddy [41] in the original paper. We experimented by randomly sampling p% from the training and validation dataset without replacement and trained the models using the sampled data. The entire test dataset was used during evaluation to ensure that the results across different p values were comparable. This process was repeated 10 times for each p value and both datasets. Figure 14 and Figure 15 show the results for the PhysioNet Challenge 2012 and MIMIC-III dataset across different p values respectively.

There is an overall decline in the performance across all models, as the models are trained on less data. However, STraTS-mTAND still generally outperforms all the models for all values of p. Hence, the addition of the mTAND module not only improved the performance of STraTS, it also did not sacrifice the generalisation ability of STraTS.

4.3.3 Self-Supervision Training. In [41], the authors proposed self-supervised forecasting training before training the model on classification to improve the performance of mortality prediction. Following the forecasting task

Table 5. Mortality prediction performance dataset averaged over 10 runs

Dataset	Model	ROC-AUC	PR-AUC
PhysioNet Challenge 2012	GRU-D	0.846 ± 0.001	0.497 ± 0.009
	SeFT	0.846 ± 0.004	0.506 ± 0.010
	mTAND	0.844 ± 0.001	0.499 ± 0.004
	STraTS	0.850 ± 0.003	0.510 ± 0.007
	STraTS-mTAND	**0.860 ± 0.004**	**0.520 ± 0.014**
MIMIC-III	GRU-D	0.870 ± 0.002	0.494 ± 0.006
	SeFT	0.870 ± 0.002	**0.499 ± 0.006**
	mTAND	0.855 ± 0.004	0.452 ± 0.018
	STraTS	0.868 ± 0.003	0.478 ± 0.010
	STraTS-mTAND	**0.871 ± 0.002**	0.493 ± 0.014

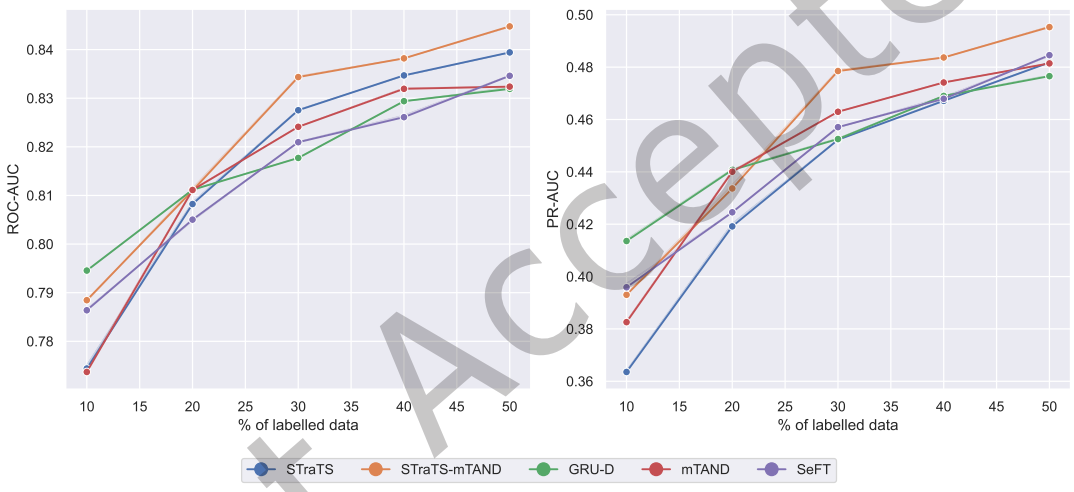

Fig. 14. Average mortality prediction performance using different percentages of labelled data over 10 runs for PhysioNet Challenge 2012. The lines show the performance of each model when trained using different percentages of training data. It shows that STraTS-mTAND generally performs the best for all percentages.

proposed there, we train the model to predict the values of the time-series variable after an observation window. For the PhysioNet Challenge 2012, the observation window is defined as $\{[0, x)|12 \leq x \leq 44, x\%4 = 0\}$, while for the MIMIC-III dataset, it is defined as $\{[0, x)|12 \leq x \leq 20, x\%4 = 0\}$. The target for each time-series variable is set as the first observation within 2 hours after the observation window. Variables that are not observed within 2 hours are masked out during the computation of the loss function.

We tested the impact of self-supervision training on STraTS-mTAND and STraTS using the PhysioNet Challenge 2012 dataset. Table 6 shows the results of the mortality prediction performance averaged over 10 runs with the best performance in **bold**. Unlike the STraTS model, the STraTS-mTAND does not seem to benefit from the self-supervision training. A hypothesis on why this may be the case could be due to the length of the time query

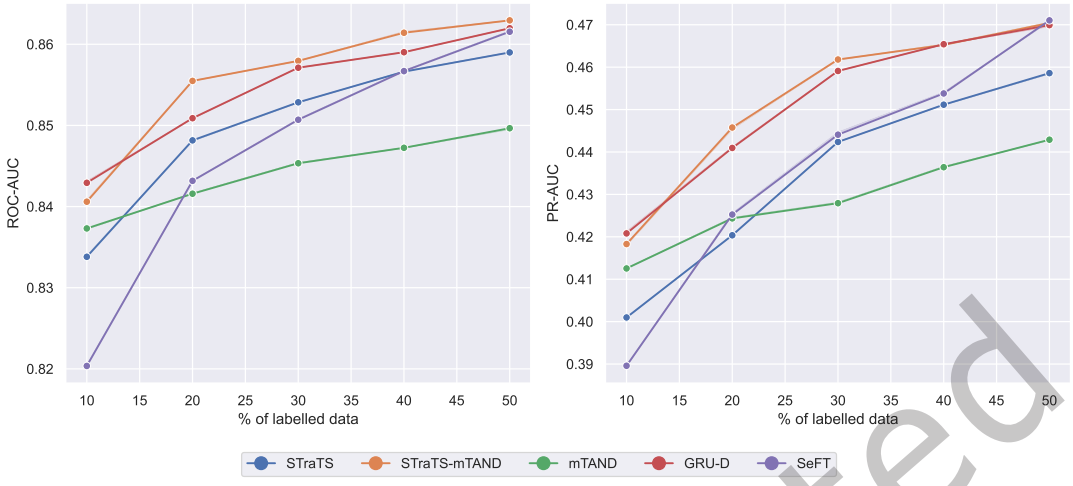

Fig. 15. Average mortality prediction performance using different percentages of labelled data over 10 runs for MIMIC-III. The lines show the performance of each model when trained using different percentages of training data. It shows that STraTS-mTAND generally performs the best for all percentages.

Table 6. Mortality prediction performance with and without self-supervision training

Model	Self-Supervision	ROC-AUC	PR-AUC
STraTS-mTAND	✓	0.855 ± 0.004	0.511 ± 0.007
	×	**0.860 ± 0.004**	**0.520 ± 0.014**
STraTS	✓	0.855 ± 0.003	0.504 ± 0.006
	×	0.850 ± 0.003	0.510 ± 0.007

n^q. In our implementation of the time query t^q, the time interval between each time point is defined as

$$t^q_{i+1} - t^q_i = \frac{\text{Length of Observation Window (Hours)}}{n^q}.$$

This time interval is constant in the mortality prediction dataset with the observation window being 24 or 48 hours. However, in the self-supervised forecasting dataset, the observation window varies across instances and ranges from 12 to 44 hours, thus the time interval also varies. This creates more variability for the model to learn. Further work can be done on testing different variations of the time query such as having time queries of different lengths for each instance or having time queries of different time intervals.

4.3.4 Impact of Time-series Sparsity. As the mTAND module performs interpolation on the input data, we expect the STraTS-mTAND to be able to hold its performance when trained on a sparser dataset. To test the robustness of STraTS-mTAND, we randomly removed timestamps and the associated observations iteratively such that only p proportion of timestamps are left in each time-series to create a sparser time-series compared to the original dataset. The above manipulations were done on training, validation and test datasets of the PhysioNet Challenge 2012 dataset. Table 7 shows the aggregated statistics of the datasets after performing the above manipulations. $p = 1.0$ refers to the original dataset.

Table 7. Statistics of manipulated PhysioNet Challenge 2012 dataset

Dataset	p	Avg time between observations(Hours)	Median # of observations	Median # of timestamps
PhysioNet Challenge 2012	1.0	0.638	424	72
	0.8	0.800	337	57
	0.6	1.062	251	42
	0.5	1.227	207	35
	0.4	1.596	163	28

The results in Table 8 and Figure 16 show the average results over 10 runs. For all values of p, the STraTS-mTAND outperforms STraTS across all variables. We also observed that for the PR-AUC metric, the mTAND model can hold its performance better as compared to STraTS, as it has a similar PR-AUC score for $p \in \{0.6, 0.4\}$ despite having a lower PR-AUC score at $p = 1.0$. Hence this suggests that the interpolations help to estimate the patients condition during long intervals without any observations and provide the classification layer with useful information for the prediction problem.

Table 8. Mortality prediction performance with datasets of different sparsity

p	Model	ROC-AUC	PR-AUC
1.0	STraTS	0.850 ± 0.003	0.510 ± 0.007
	mTAND	0.844 ± 0.001	0.499 ± 0.004
	STraTS-mTAND	0.860 ± 0.004	0.520 ± 0.014
0.8	STraTS	0.853 ± 0.003	0.507 ± 0.006
	mTAND	0.846 ± 0.002	0.503 ± 0.003
	STraTS-mTAND	0.860 ± 0.004	0.519 ± 0.014
0.6	STraTS	0.843 ± 0.005	0.489 ± 0.017
	mTAND	0.839 ± 0.002	0.489 ± 0.007
	STraTS-mTAND	0.853 ± 0.003	0.515 ± 0.008
0.5	STraTS	0.837 ± 0.004	0.492 ± 0.011
	mTAND	0.830 ± 0.002	0.479 ± 0.003
	STraTS-mTAND	0.844 ± 0.003	0.499 ± 0.011
0.4	STraTS	0.836 ± 0.006	0.473 ± 0.015
	mTAND	0.826 ± 0.004	0.494 ± 0.009
	STraTS-mTAND	0.841 ± 0.004	0.472 ± 0.004

5 ICU MORTALITY STUDY

Using our STraTS-mTAND model, we delve into the ICU Mortality predictability problem. Our focus lies in comprehending the influence of altering the prediction time window on prediction accuracy, as well as the impact of incorporating static (demographic) information into the prediction process. Both problems are crucial factors in clinical decision-making and patient management within ICUs.

In clinical practice, the ability to predict patient outcomes within specific time frames is vital for timely interventions and resource allocation [30]. For instance, predicting mortality risk within the next 12 hours allows

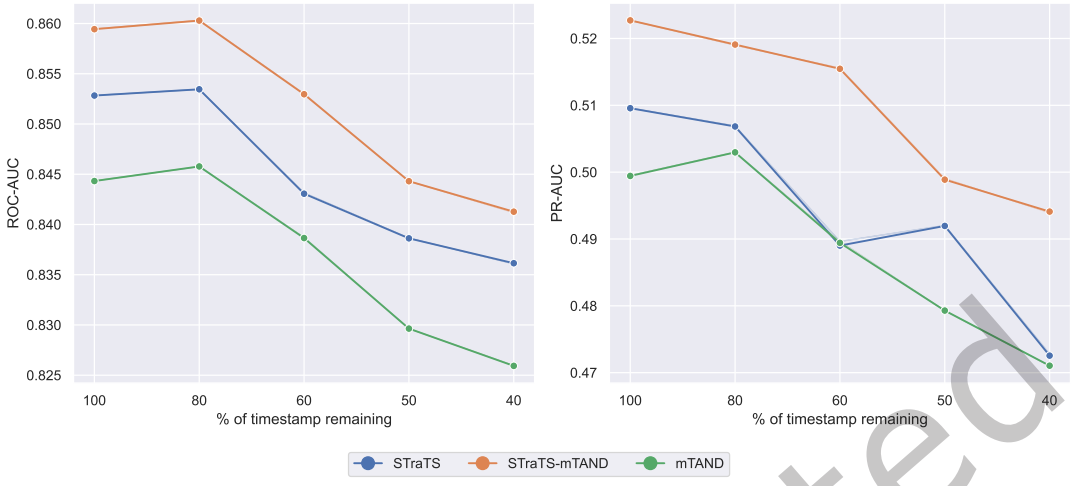

Fig. 16. The average mortality prediction performance across datasets with varying levels of sparsity. Each line represents the performance of individual models trained using datasets with different p values. It is evident that STraTS-mTAND consistently outperforms other models across all p values, demonstrating its superior predictive capability regardless of data sparsity.

healthcare providers to prioritize care for critically ill patients who require immediate attention. Similarly, longer prediction time windows, such as 24 hours or longer, enable clinicians to anticipate patient deterioration and plan interventions accordingly. Therefore, understanding how changing the prediction time window affects prediction accuracy provides valuable insights into the optimal timing of clinical interventions and improves patient outcomes [34].

Incorporating static demographic information, such as age, gender, and comorbidity, into mortality prediction models are believed to enhance their predictive accuracy and clinical utility [23]. Demographic factors play a significant role in determining patient susceptibility to certain conditions and treatment outcomes [40]. For example, advanced age and certain comorbidity may increase the risk of mortality or complications during hospitalization. By considering these factors in mortality prediction models, clinicians can better tailor treatment plans, allocate resources effectively, and provide personalized care to high-risk patients. Understanding the impact of static demographic information on prediction accuracy is therefore essential for developing robust and clinically relevant predictive models in ICU settings [3].

5.1 Prediction Time Window Analysis

To try and identify which time window provides the most explanation on the time-series, we manipulated the dataset to filter for observations within a certain time window. Table 17 (appendix) shows the aggregated statistics of the three datasets after filtering for the respective time window. Table 9 shows the results according to the different time intervals. The **best** performance is bold and underlined, **2nd best** is bold and 3rd best performance is underlined.

When the model is trained on 12 consecutive hours of data, using the time intervals 0-12 hours or 12-24 hours appears to yield poorer performance compared to utilizing 24-36 hours or 36-48 hours, as illustrated in Figure 17. Notably, this discrepancy persists despite the approximate equality in the number of observations across each interval. Such findings suggest that observations collected at later time points hold greater significance than those

Table 9. STraTS-mTAND mortality prediction performance using different time interval

Starting Hour	Ending Hour	ROC-AUC	PR-AUC
0	12	0.805 ± 0.003	0.399 ± 0.015
12	24	0.800 ± 0.003	0.420 ± 0.007
0	24	0.828 ± 0.005	0.445 ± 0.007
24	36	0.817 ± 0.002	0.467 ± 0.006
12	36	0.829 ± 0.002	0.481 ± 0.004
0	36	**0.849 ± 0.004**	0.491 ± 0.013
36	48	0.818 ± 0.004	0.469 ± 0.009
24	48	0.842 ± 0.003	0.514 ± 0.009
12	48	0.843 ± 0.003	**0.519 ± 0.006**
0	48	**0.860 ± 0.004**	**0.520 ± 0.014**

obtained earlier. Conversely, when the model is trained on 24 consecutive hours of data, depicted in Figure 18, a notable disparity in performance arises when using the 24-48 hours interval compared to other time windows. This aligns with our observations from Figure 17, wherein the 24-36 hours and 36-48 hours intervals outperform the 0-12 hours and 12-24 hours intervals. Moreover, the observed enhancement in performance upon combining the 24-36 hours and 36-48 hours intervals implies that the information provided by these two time windows is complementary and augments the model's predictive capacity. Figure 19 exhibits the performance results across increasing time windows, ranging from 0 hours to 48 hours. As anticipated, there is a discernible uptrend in performance due to the increased number of observations per time-series. Similarly, Figure 20 showcases the performance trend with increasing time windows, this time starting from 48 hours to 0 hours. As expected, a corresponding upward trend is observed. However, a notable finding arises when comparing the performance of the 12-48 hours and 24-48 hours intervals, where the performance increase is marginal. Conversely, a more substantial performance enhancement is noted upon subsequent inclusion of the 0-12 hours time window.

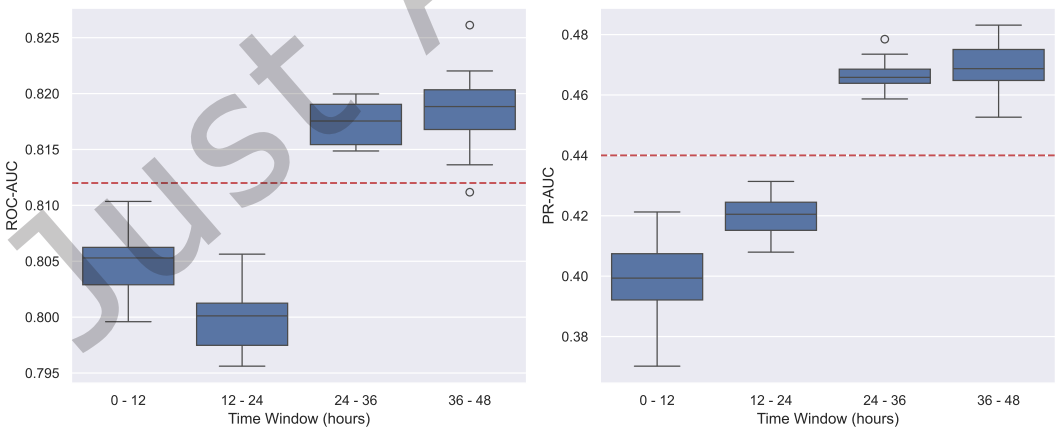

Fig. 17. Boxplot of mortality prediction performance using 12 hours data measured with ROC-AUC and PR-AUC. There is a clear distinction of the ROC-AUC and PR-AUC when comparing 0-12, 12-24 and 24-36, 36-48 observation windows.

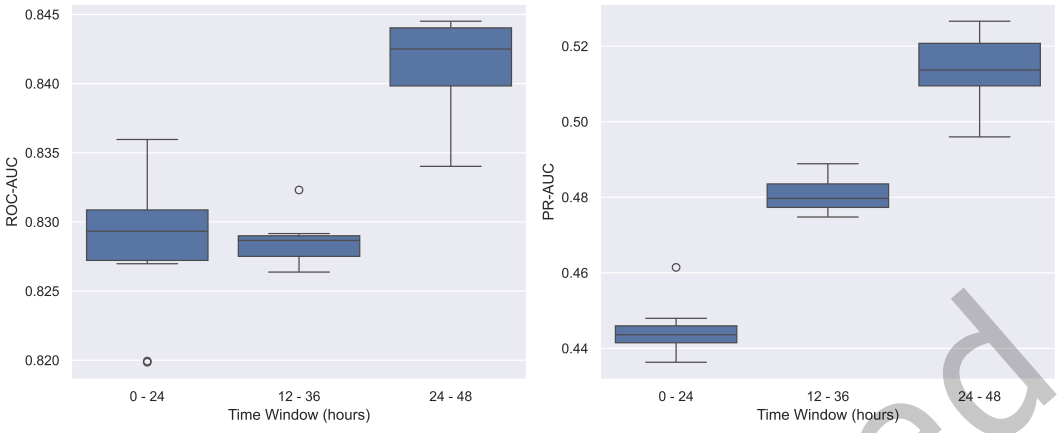

Fig. 18. Boxplot of mortality prediction performance using 24 hours data. There is a clear distinction of the ROC-AUC and PR-AUC when comparing 0-24, 12-36 and 24-48 observation windows.

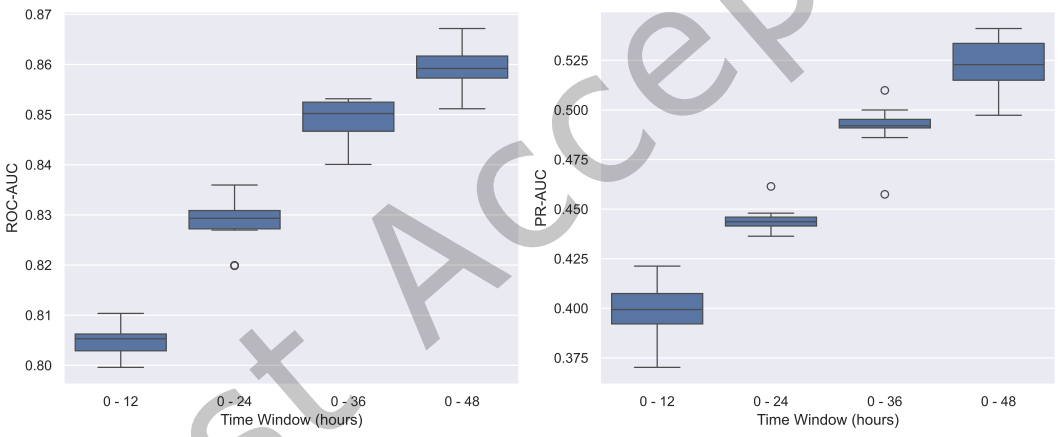

Fig. 19. Mortality prediction performance with increasing time window, starting from 0 hours to 48 hours. The end of the observation window increases from 12 hours to 48 hours, with an increase of 12 hours each. The ROC-AUC and PR-AUC score increases with a longer observation window.

From the findings presented in Figure 20, we proceeded to further investigation by training the model using time windows of 0-12 hours and 24-48 hours. Table 10 indicates that excluding data from the 12-24 hours interval results in comparable performance to using the entire dataset. This suggests that the information provided by the 0-12 hours time window alone may not be sufficiently informative, as the observations are not recent. However, integrating data from the 24-48 hours time window leads to a significant enhancement in performance. This underscores the notion that not all observations contribute equally to the prediction problem. We hypothesize that closer time intervals between observations do not necessarily offer a comprehensive overview of the trajectory of

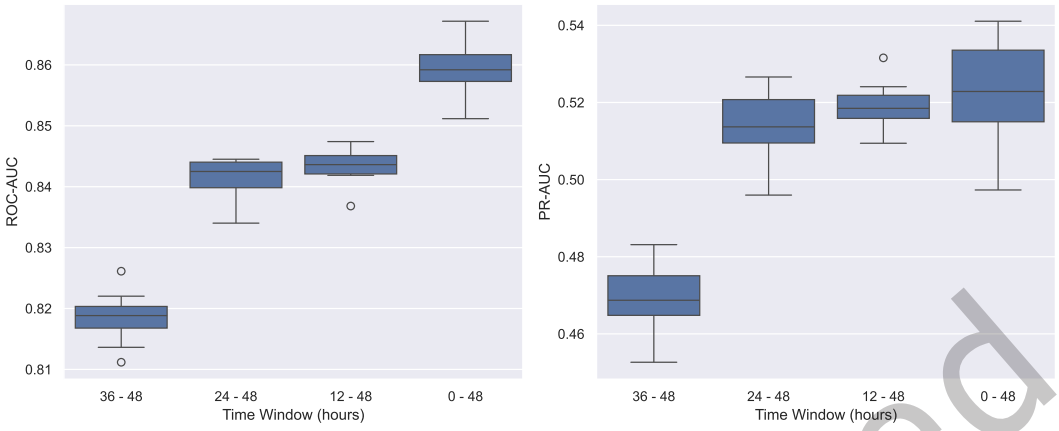

Fig. 20. Mortality prediction performance with increasing time window, starting from 48 hours to 0 hours. The start of the observation window decreases from 36 hours to 0 hours, with a decrease of 12 hours each. The ROC-AUC and PR-AUC score increases with a longer observation window. However, the performance increase when comparing 36-48 hours against 24-48 hours is negligible.

a patient's condition. Instead, there exists an optimal time interval that allows the model to effectively estimate the trajectory, and any observations taken at closer intervals may not necessarily improve estimation accuracy.

Table 10. STraTS-mTAND mortality prediction performance

Time Window(Hours)	ROC-AUC	PR-AUC
0-12 & 24-48	0.858 ± 0.002	**0.525 ± 0.010**
0-48	**0.860 ± 0.004**	0.520 ± 0.014

5.2 Static Variable Analysis

We removed the static variables from the dataset to try to understand how the static variables are affecting the model prediction abilities. We obtained the results by averaging the results over 10 runs. The results are shown in Table 11 with the **best** performance in bold. Without static variables, the ROC-AUC declined by 1.1%. This suggests to us that the static variables do aid in the prediction task in general.

Table 11. Mortality prediction performance with and without static variables averaged over 10 runs.

	ROC-AUC	PR-AUC
With Static Variables	**0.860 ± 0.004**	**0.520 ± 0.014**
Without Static Variables	0.849 ± 0.003	0.504 ± 0.011

To assess the impact of static variables on predicted probabilities, we analyzed the density distribution of predicted probabilities for both models across different variables. Figure 21 illustrates the probability distribution

categorized by age, with each group comprising 25% of the test dataset. Notably, for younger patients aged 14 to 52, the model trained with static variables tends to predict a lower probability of mortality compared to the model trained without static variables. Conversely, for older patients aged 78 to 90, the probability of mortality tends to be higher with the model trained with static variables. This observation aligns with the intuition that younger patients are generally more resilient and have a higher likelihood of survival given similar medical conditions.

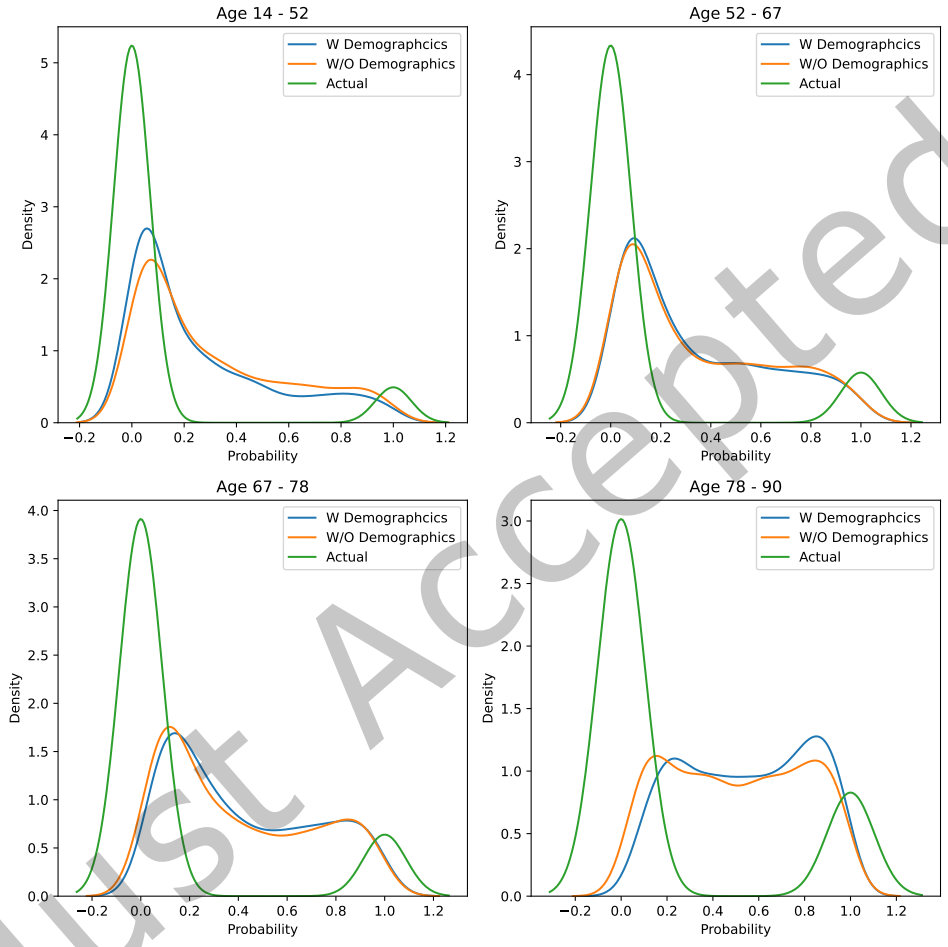

Fig. 21. Average Predicted Probability from models with and without static variables is compared in the following visualization. The probability distribution of output predictions by models trained with and without static variables is grouped into four age buckets. In the youngest age bucket, the model trained with static variables exhibits a higher density at lower probabilities. Conversely, in the oldest age bucket, the model trained with static variables shows a higher density at higher probabilities.

Similarly, Figure 22 illustrates the probability distribution categorized by gender, showing that both males and females will have a higher likelihood of mortality in the model trained with static variables compared with the model trained without static variables, better reflecting the actual class distribution for each gender. Table 12 shows that although the gender variable will not actually affect ICU mortality, it will make a difference in

the predictions. The results shown in Table 12 are based on the chosen dataset, which may not be a general conclusion.

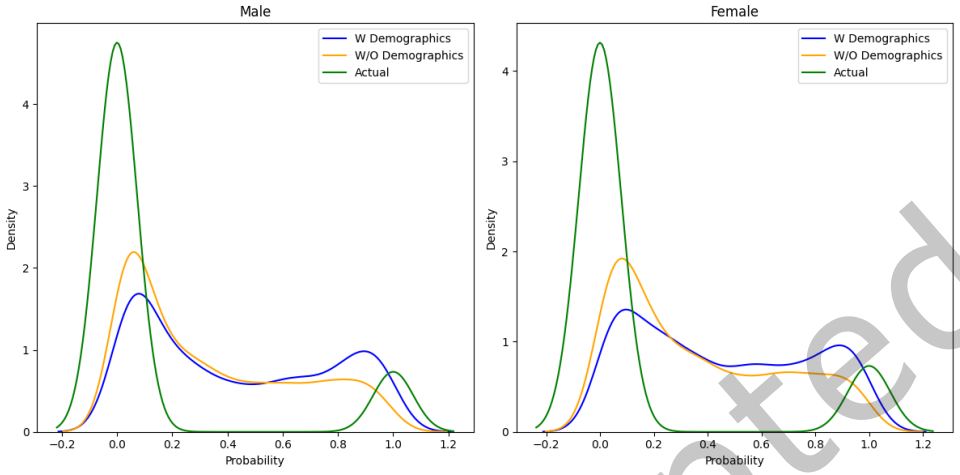

Fig. 22. Average Predicted Probability from models with and without static variables is compared in the following visualization. The probability distribution of output predictions by models trained with and without static variables is grouped into two gender buckets. In the male bucket, the model trained with static variables exhibits a higher density at higher probabilities. Conversely, in the female bucket, the model trained with static variables shows a higher density at lower probabilities.

Table 12. Gender-based comparison of Actual and Predicted Results. The mean is the numeric average of prediction probabilities in each data group and the p-value is obtained by t-test.

	Gender	Mean	p-value
Actual	Male	0.134	0.309
	Female	0.145	
With demographics	Male	0.427	0.016
	Female	0.452	
Without demographics	Male	0.338	0.010
	Female	0.363	

Further exploration into the probability densities, broken down by ICU types as depicted in Figure 23, reveals that models trained with static variables tend to better reflect the actual class distribution of each ICU type. For instance, in the Cardiac Surgery Recovery Unit, where a higher proportion of in-hospital survival (class 0) is observed compared to other ICU types, the model trained with static variables exhibits a higher density at lower probabilities (e.g., 0.1) and lower densities at higher probabilities (e.g., 0.7 and above). We also show the p-value in this setting in Table 13, which is obtained by one-way ANOVA. The results indicate that the ICU types have a significant impact on ICU mortality, where Medical ICU will have a higher probability of mortality. The results shown in Table 13 are based on the chosen dataset, which may not be a general conclusion.

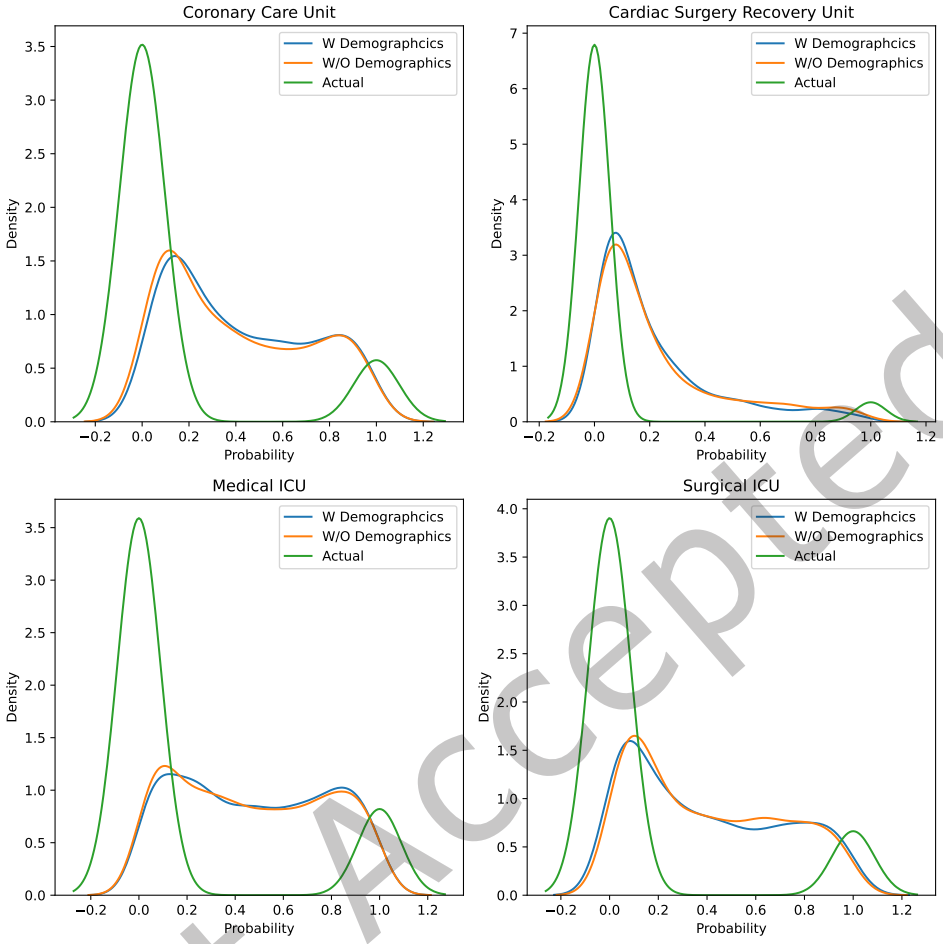

Fig. 23. Average Predicted Probability from models trained with and without static variables. Models trained with static variables tend to fit the actual class distribution of each ICU type slightly better.

Overall, while models trained without static variables are capable of generating similar probability distributions, the inclusion of static variables proves advantageous. These variables subtly shift the probability distribution to better align with the actual class distribution, resulting in improved performance, as evidenced by the results in Table 11. This underscores the utility of static variables in refining prediction models and enhancing their predictive accuracy, although the improvement may not be as significant as one has expected.

6 CONCLUSION

In recent years, various techniques have emerged to harness irregularly sampled time-series data for predictive modeling in the medical domain. These approaches typically fall into two broad categories: interpolation-based and non-interpolation-based methods, each offering distinct advantages and limitations. In this study, we introduced a novel model, STraTS-mTAND, which integrates STraTS, a non-interpolation-based model, with mTAND, an

Table 13. ICU type-based comparison of Actual and Predicted Results. The mean is the numeric average of prediction probabilities in each data group and the p-value is obtained by one-way ANOVA.

	ICU Type	Mean	p-value
Actual	Coronary Care Unit	0.140	$6.93e^{-19}$
	Cardiac Surgery Recovery Unit	0.049	
	Medical ICU	0.186	
	Surgical ICU	0.145	
With demographics	Coronary Care Unit	0.498	$1.49e^{-126}$
	Cardiac Surgery Recovery Unit	0.221	
	Medical ICU	0.540	
	Surgical ICU	0.441	
Without demographics	Coronary Care Unit	0.372	$4.33e^{-79}$
	Cardiac Surgery Recovery Unit	0.188	
	Medical ICU	0.429	
	Surgical ICU	0.357	

interpolation-based model. By combining these techniques, our model aims to provide a more comprehensive representation of the time-series data and enhance predictive performance.

Through experimentation with the PhysioNet Challenge 2012 and MIMIC-III datasets, we evaluated the efficacy of our model in the context of in-hospital mortality prediction. Our results demonstrate that STraTS-mTAND consistently outperforms other interpolation and non-interpolation techniques in terms of key metrics such as ROC-AUC and PR-AUC. Importantly, our model exhibits robust performance even when trained with limited data and sparser time-series datasets, highlighting its versatility and applicability in real-world clinical settings.

Our study revealed that selecting appropriate time windows significantly influences predictive performance in analyzing irregularly sampled time-series data. Specifically, observations collected at later time points demonstrated greater significance in mortality prediction. Additionally, the inclusion of static variables subtly enhanced predictive accuracy by better aligning the probability distribution with actual class distributions. These findings underscore the importance of thoughtful consideration of time windows and static variables in predictive modeling for improved clinical decision-making.

Moving forward, there are several avenues for further research and development. One area of interest is the exploration of different variations of the time query in the mTAND module. Our experiments utilized fixed-length and regular intervals for the time query, but investigating alternative configurations could provide valuable insights into how varying lengths and intervals impact model performance. Additionally, future studies may explore the incorporation of additional features or data sources to further enhance predictive accuracy and broaden the scope of applications.

Overall, we believe that our work contributes to advancing predictive analytics in healthcare and holds the potential to positively impact clinical decision-making and patient care. By accurately identifying high-risk patients and facilitating timely interventions, our model has the potential to improve patient outcomes and ultimately save lives. We hope that our findings will inspire further research in this field and pave the way for the development of more effective predictive models in healthcare.

ACKNOWLEDGMENTS

This research is supported by the Ministry of Education, Singapore. (Grant ID: RG17/22, RS15/23)

REFERENCES

[1] Julia Adler-Milstein, Catherine M DesRoches, Peter Kralovec, Gregory Foster, Chantal Worzala, Dustin Charles, Talisha Searcy, and Ashish K Jha. 2015. Electronic Health Record Adoption In US Hospitals: Progress Continues, But Challenges Persist. *Health Aff (Millwood)* 34, 12 (Nov. 2015), 2174–2180.

[2] Ane Blázquez-García, Kristoffer Wickstrøm, Shujian Yu, Karl Øyvind Mikalsen, Ahcène Boubekki, Angel Conde, Usue Mori, Robert Jenssen, and Jose A. Lozano. 2023. Selective Imputation for Multivariate Time Series Datasets With Missing Values. *IEEE Transactions on Knowledge and Data Engineering* 35, 9 (Sept. 2023), 9490–9501. https://doi.org/10.1109/TKDE.2023.3240858

[3] Ricardo MS Carvalho, Daniela Oliveira, and Catia Pesquita. 2023. Knowledge Graph Embeddings for ICU readmission prediction. *BMC Medical Informatics and Decision Making* 23, 1 (2023), 12.

[4] Vinod Kumar Chauhan, Anshul Thakur, Odhran O'Donoghue, Omid Rohanian, Soheila Molaei, and David A. Clifton. 2024. Continuous Patient State Attention Model for Addressing Irregularity in Electronic Health Records. *BMC Medical Informatics and Decision Making* 24, 1 (May 2024), 117. https://doi.org/10.1186/s12911-024-02514-2

[5] 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 (April 2018), 6085.

[6] Yang Chen, Hong Liu, Pinhao Song, and Wenhao Li. 2024. Neural Ordinary Differential Equation for Irregular Human Motion Prediction. *Pattern Recognition Letters* 178 (Feb. 2024), 76–83. https://doi.org/10.1016/j.patrec.2023.12.016

[7] Yuqi Chen, Kan Ren, Yansen Wang, Yuchen Fang, Weiwei Sun, and Dongsheng Li. 2024. ContiFormer: Continuous-Time Transformer for Irregular Time Series Modeling. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (Nips '23)*. Curran Associates Inc., Red Hook, NY, USA, Article 2042.

[8] Yu-wen Chen, Yu-jie Li, Peng Deng, Zhi-yong Yang, Kun-hua Zhong, Li-ge Zhang, Yang Chen, Hong-yu Zhi, Xiao-yan Hu, Jian-teng Gu, et al. 2022. Learning to predict in-hospital mortality risk in the intensive care unit with attention-based temporal convolution network. *BMC anesthesiology* 22, 1 (2022), 119.

[9] Chih-Chou Chiu, Chung-Min Wu, Te-Nien Chien, Ling-Jing Kao, Chengcheng Li, and Chuan-Mei Chu. 2023. Integrating structured and unstructured EHR data for predicting mortality by machine learning and latent Dirichlet allocation method. *International Journal of Environmental Research and Public Health* 20, 5 (2023), 4340.

[10] Ranak Roy Chowdhury, Jiacheng Li, Xiyuan Zhang, Dezhi Hong, Rajesh K. Gupta, and Jingbo Shang. 2023. PrimeNet: Pre-training for Irregular Multivariate Time Series. *Proceedings of the AAAI Conference on Artificial Intelligence* 37, 6 (June 2023), 7184–7192. https://doi.org/10.1609/aaai.v37i6.25876

[11] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. arXiv:1412.3555 [cs.NE]

[12] Federico Cismondi, André S. Fialho, Susana M. Vieira, Shane R. Reti, João M.C. Sousa, and Stan N. Finkelstein. 2013. Missing data in medical databases: Impute, delete or classify? *Artificial Intelligence in Medicine* 58, 1 (2013), 63–72. https://doi.org/10.1016/j.artmed.2013.01.003

[13] Sajad Darabi, Mohammad Kachuee, Shayan Fazeli, and Majid Sarrafzadeh. 2020. Taper: Time-aware patient ehr representation. *IEEE journal of biomedical and health informatics* 24, 11 (2020), 3268–3275.

[14] Jesse Davis and Mark Goadrich. 2006. The Relationship Between Precision-Recall and ROC Curves. *Proceedings of the 23rd International Conference on Machine Learning, ACM* 06. https://doi.org/10.1145/1143844.1143874

[15] Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch, and Stephan Mandt. 2020. GP-VAE: Deep Probabilistic Time Series Imputation. arXiv:1907.04155 [stat.ML]

[16] Karimollah Hajian-Tilaki. 2013. Receiver Operating Characteristic (ROC) Curve Analysis for Medical Diagnostic Test Evaluation. *Caspian J Intern Med* 4, 2 (2013), 627–635.

[17] Hrayr Harutyunyan, Hrant Khachatrian, David C. Kale, Greg Ver Steeg, and Aram Galstyan. 2019. Multitask learning and benchmarking with clinical time series data. *Scientific Data* 6, 1 (2019), 96. https://doi.org/10.1038/s41597-019-0103-9

[18] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[19] Max Horn, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. 2020. Set Functions for Time Series. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, 4353–4363.

[20] George Hripcsak, David J Albers, and Adler Perotte. 2015. Parameterizing time in electronic health record studies. *Journal of the American Medical Informatics Association* 22, 4 (2015), 794–804.

[21] Lynette A Hunt. 2017. Missing data imputation and its effect on the accuracy of classification. In *Data Science: Innovative Developments in Data Analysis and Clustering*. Springer, 3–14.

[22] Shinya Iwase, Taka-aki Nakada, Tadanaga Shimada, Takehiko Oami, Takashi Shimazui, Nozomi Takahashi, Jun Yamabe, Yasuo Yamao, and Eiryo Kawakami. 2022. Prediction algorithm for ICU mortality and length of stay using machine learning. *Scientific reports* 12, 1 (2022), 12912.

[23] Fanny Janssen. 2018. Advances in mortality forecasting: introduction. *Genus* 74, 1 (2018), 21.

[24] Alistair E W Johnson, Tom J Pollard, Lu Shen, Li-Wei H Lehman, 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, 1 (May 2016), 160035.

[25] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG]

[26] Zachary C Lipton, David Kale, and Randall Wetzel. 2016. Directly Modeling Missing Data in Sequences with RNNs: Improved Classification of Clinical Time Series. In *Proceedings of the 1st Machine Learning for Healthcare Conference (Proceedings of Machine Learning Research, Vol. 56)*, Finale Doshi-Velez, Jim Fackler, David Kale, Byron Wallace, and Jenna Wiens (Eds.). PMLR, Northeastern University, Boston, MA, USA, 253–270. https://proceedings.mlr.press/v56/Lipton16.html

[27] Roderick JA Little and Donald B Rubin. 2019. *Statistical analysis with missing data*. Vol. 793. John Wiley & Sons.

[28] Alessandra Lunardi. 2018. *Interpolation theory*. Vol. 16. Springer.

[29] Matthew McDermott, Bret Nestor, Evan Kim, Wancong Zhang, Anna Goldenberg, Peter Szolovits, and Marzyeh Ghassemi. 2021. A comprehensive EHR timeseries pre-training benchmark. In *Proceedings of the Conference on Health, Inference, and Learning* (Virtual Event, USA) *(CHIL '21)*. Association for Computing Machinery, New York, NY, USA, 257–278. https://doi.org/10.1145/3450439.3451877

[30] Hendrik-Jan Mijderwijk and Hans-Jakob Steiger. 2022. Predictive analytics in clinical practice: advantages and disadvantages. In *Machine learning in clinical neuroscience: foundations and applications*. Springer, 263–268.

[31] Manel Mili, Asma Kerkeni, Asma Ben Abdallah, and Mohamed Hedi Bedoui. 2022. ICU Mortality Prediction Using Long Short-Term Memory Networks. In *International Conference on Intelligent Data Engineering and Automated Learning*. Springer, 242–251.

[32] Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. 2016. Phased LSTM: Accelerating Recurrent Network Training for Long or Event-based Sequences. arXiv:1610.09513 [cs.LG]

[33] Trang Pham, Truyen Tran, Dinh Phung, and Svetha Venkatesh. 2017. Predicting healthcare trajectories from medical records: A deep learning approach. *Journal of Biomedical Informatics* 69 (2017), 218–229. https://doi.org/10.1016/j.jbi.2017.04.001

[34] Sarah Pungitore and Vignesh Subbian. 2023. Assessment of Prediction Tasks and Time Window Selection in Temporal Modeling of Electronic Health Record Data: a Systematic Review. *Journal of Healthcare Informatics Research* 7, 3 (2023), 313–331.

[35] Mohammad Reza Rezaei-Dastjerdehei, Amirmohammad Mijani, and Emad Fatemizadeh. 2020. Addressing Imbalance in Multi-Label Classification Using Weighted Cross Entropy Loss Function. In *2020 27th National and 5th International Iranian Conference on Biomedical Engineering (ICBME)*. 333–338. https://doi.org/10.1109/ICBME51989.2020.9319440

[36] Mohammed Saeed, Mauricio Villarroel, Andrew T Reisner, Gari Clifford, Li-Wei Lehman, George Moody, Thomas Heldt, Tin H Kyaw, Benjamin Moody, and Roger G Mark. 2011. Multiparameter Intelligent Monitoring in Intensive Care II: a public-access intensive care unit database. *Crit Care Med* 39, 5 (May 2011), 952–960.

[37] Takaya Saito and Marc Rehmsmeier. 2015. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLoS One* 10, 3 (March 2015), e0118432.

[38] Satya Narayan Shukla and Benjamin Marlin. 2021. Multi-Time Attention Networks for Irregularly Sampled Time Series. In *International Conference on Learning Representations*. https://openreview.net/forum?id=4c0J6lwQ4_

[39] Ikaro Silva, George Moody, Daniel J Scott, Leo A Celi, and Roger G Mark. 2012. Predicting In-Hospital Mortality of ICU Patients: The PhysioNet/Computing in Cardiology Challenge 2012. *Comput Cardiol (2010)* 39 (2012), 245–248.

[40] Daniel Stoessel, Rui Fa, Svetlana Artemova, Ursula von Schenck, Hadiseh Nowparast Rostami, Pierre-Ephrem Madiot, Caroline Landelle, Fréderic Olive, Alison Foote, Alexandre Moreau-Gaudry, et al. 2023. Early prediction of in-hospital mortality utilizing multivariate predictive modelling of electronic medical records and socio-determinants of health of the first day of hospitalization. *BMC Medical Informatics and Decision Making* 23, 1 (2023), 259.

[41] 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 (7 2022), 17 pages. https://doi.org/10.1145/3516367

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. arXiv:1706.03762 [cs.CL]

[43] Philip B. Weerakody, Kok Wai Wong, Guanjin Wang, and Wendell Ela. 2021. A review of irregular time series data handling with gated recurrent neural networks. *Neurocomputing* 441 (2021), 161–178. https://doi.org/10.1016/j.neucom.2021.02.046

[44] Hyowon Wi, Yehjin Shin, and Noseong Park. 2024. Continuous-Time Autoencoders for Regular and Irregular Time Series Imputation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*. ACM, Merida Mexico, 826–835. https://doi.org/10.1145/3616855.3635831

[45] Yanbo Xu, Shangqing Xu, Manav Ramprassad, Alexey Tumanov, and Chao Zhang. 2023. TransEHR: Self-Supervised Transformer for Clinical Time Series Data. In *Proceedings of the 3rd Machine Learning for Health Symposium*. PMLR, 623–635.

[46] Pranjul Yadav, Michael Steinbach, Vipin Kumar, and Gyorgy Simon. 2018. Mining Electronic Health Records (EHRs): A Survey. *ACM Comput. Surv.* 50, 6, Article 85 (1 2018), 40 pages. https://doi.org/10.1145/3127881

[47] Lin Zehui, Pengfei Liu, Luyao Huang, Junkun Chen, Xipeng Qiu, and Xuanjing Huang. 2019. DropAttention: A Regularization Method for Fully-Connected Self-Attention Networks. arXiv:1907.11065 [cs.CL]

[48] Yi Zheng and Xiangpei Hu. 2020. Healthcare predictive analytics for disease progression: a longitudinal data fusion approach. *Journal of Intelligent Information Systems* 55, 2 (2020), 351–369.
[49] Xi Zhou, Wei Xiang, and Tao Huang. 2023. A Novel Neural Network for Improved In-Hospital Mortality Prediction with Irregular and Incomplete Multivariate Data. *Neural Networks* 167 (Oct. 2023), 741–750. https://doi.org/10.1016/j.neunet.2023.07.033

A ADDITIONAL TABLES

Table 14. Hyperparameters of Baseline Models

Model	PhysioNet Challenge 2012	MIMIC-III
GRU-D	n_units=49, recurrent_dropout=0.2, dropout=0.2, lr=0.0004	n_units=60, recurrent_dropout=0.2, dropout=0.2, lr=0.0004
SeFT	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	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
mTAND	d=32, he=8, dropout=0.2, len_time_query=200, lr=0.0005	d=32, he=8, dropout=0.2, len_time_query=100, lr=0.0005
STraTS	d=32, N=2, he=4, dropout=0.2, lr=0.0005	d=32, N=2, he=4, dropout=0.2, lr=0.0005
STraTS-mTAND	d_strats=32, N_strats=2, he_strats=4, dropout_strats=0.2, d_mtand=32, he_mtand=8, dropout_mtand=0.2, len_time_query=200, lr=0.0005	d_strats=32, N_strats=2, he_strats=4, dropout_strats=0.2, d_mtand=32, he_mtand=8, dropout_mtand=0.2, len_time_query=100, lr=0.0005

Table 15. Missingness of features in PhysioNet-2012 dataset

Time-series Variable	Average # of Observations per patient	% of patients with at least one observation
ALP	0.79	42.52
ALT	0.81	43.49
AST	0.81	43.49
Albumin	0.60	40.63
BUN	3.48	98.57
Bilirubin	0.82	43.49
Cholesterol	0.08	7.91
Creatinine	3.50	98.57
DiasABP	36.71	70.29
FiO2	7.94	67.70
GCS	15.51	98.56
Glucose	3.28	97.65
HCO3	3.41	98.37
HCT	4.59	98.52
HR	57.32	98.57
K	3.65	98.02
Lactate	2.03	54.85
MAP	36.78	70.14
MechVent	7.64	63.24
Mg	3.41	97.66
NIDiasABP	24.58	87.33
NIMAP	24.23	87.29
NISysABP	24.61	87.69
Na	3.42	98.33
PaCO2	5.76	75.53
PaO2	5.75	75.53
Platelets	3.56	98.45
RespRate	13.80	27.76
SaO2	2.00	44.78
SysABP	36.72	70.30
Temp	21.64	98.56
TroponinI	0.10	4.71
TroponinT	0.52	22.00
Urine	34.30	97.51
WBC	3.25	98.32
Weight	32.19	92.43
pH	6.02	75.98

Table 16. Missingness of features in MIMIC-III dataset

Time-series Variable	Average # of Observations per patient	% of patients with at least one observation
Capillary refill rate	0.08	1.23
Diastolic blood pressure	29.30	98.80
Fraction inspired oxygen	2.15	25.00
Glascow coma scale eye opening	7.81	99.05
Glascow coma scale motor response	7.77	99.02
Glascow coma scale total	4.41	55.21
Glascow coma scale verbal response	7.78	99.01
Glucose	7.71	99.22
Heart Rate	31.08	98.80
Height	0.18	17.48
Mean blood pressure	29.10	98.79
Oxygen saturation	31.44	99.23
Respiratory rate	31.31	98.70
Systolic blood pressure	29.31	98.80
Temperature	9.80	97.27
Weight	1.31	67.81
pH	3.93	69.84

Table 17. Statistics of filtered dataset

Starting Hour	Ending Hour	Avg time between observations(Hours)	Median # of observations	Median # of timestamps
0	12	0.491	123	22
12	24	0.633	101	17
0	24	0.562	228	40
24	36	0.695	96	15
12	36	0.674	199	33
0	36	0.607	329	56
36	48	0.727	91	15
24	48	0.723	189	31
12	48	0.699	294	48
0	48	0.638	424	72