Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Studies of Phenotypes and Clinical Applications

Temporal Convolutional Networks and Dynamic Time Warping can Drastically Improve the Early Prediction of Sepsis

Michael Moor [1,2], Max Horn [1,2], Bastian Rieck [1,2], Damian Roqueiro [1,2] and Karsten Borgwardt [1,2,*]

[1] Department of Biosystems Science and Engineering, ETH Zurich, Basel, 4058, Switzerland
[2] SIB Swiss Institute of Bioinformatics

* To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Sepsis is a life-threatening host response to infection associated with high mortality, morbidity and health costs. Its management is highly time-sensitive since each hour of delayed treatment increases mortality due to irreversible organ damage. Meanwhile, despite decades of clinical research robust biomarkers for sepsis are missing. Therefore, detecting sepsis early by utilizing the affluence of high-resolution intensive care records has become a challenging machine learning problem. Recent advances in deep learning and data mining promise a powerful set of tools to efficiently address this task.

Results: This paper proposes two approaches for the early detection of sepsis: a new deep learning model (MGP-TCN) and a data mining model (DTW-KNN). MGP-TCN employs a temporal convolutional network as embedded in a Multitask Gaussian Process Adapter framework, making it directly applicable to irregularly spaced time series data. Our DTW-KNN is an ensemble approach that employs dynamic time warping. We then frame the timely detection of sepsis as a supervised time series classification task. For this, we derive the most recent sepsis definition in an hourly resolution to provide the first fully accessible early sepsis detection environment. Seven hours before sepsis onset, our methods MGP-TCN/DTW-KNN improve area under the precision–recall curve from 0.25 to 0.35/0.40 over the state of the art. This demonstrates that they are well-suited for detecting sepsis in the crucial earlier stages when management is most effective.

Availability: We make our methods and scripts available under
`https://osf.io/av5yx/?view_only=a6e3442634b34d53ba6e59c4a956b318`

Contact: karsten.borgwardt@bsse.ethz.ch

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Sepsis describes a severe complication from infections. Despite decades of clinical research, sepsis remains a major public health issue with high mortality, morbidity, and related health costs (Dellinger *et al.*, 2013; Kaukonen *et al.*, 2014; Peake *et al.*, 2007; Hotchkiss *et al.*, 2016). Singer *et al.* (2016) define sepsis as a life-threatening organ dysfunction caused by a dysregulated host response to infection. Currently, when sepsis is detected and the underlying pathogen is identified, organ damage has already progressed to a potentially irreversible stage. Effective management is therefore highly time-sensitive and often decides about life and death in the intensive care unit (ICU). From sepsis onset, each hour of delayed effective antibiotic treatment is known to increase mortality (Ferrer *et al.*, 2014). Detecting sepsis *early* has therefore also gained considerable attention from the machine learning field (Calvert *et al.*, 2016; Desautels *et al.*, 2016; Kam and Kim, 2017; Futoma *et al.*, 2017a).

Conventionally, sepsis detection has been framed as a multi-channel time series classification task performed after applying a set of hand-crafted preprocessing steps, such as binning, carry-forward imputation, and rolling means (Calvert *et al.*, 2016; Desautels *et al.*, 2016). Adding to this, we propose DTW-KNN, a classic method consisting of an ensemble of multiple k-nearest neighbor classifiers that employ the dynamic time warping distance.

However, clinical data is typically sampled irregularly and data sparsity carries crucial information in this context. Most existing approaches nonetheless require the data to be processed to a stage where sampling information is *not* available for modelling any more, which may impede training and lead to lower predictive performance in turn. Futoma *et al.* (2017a) proposed a sepsis detection method that addresses for irregular sampling by applying the Gaussian Process Adapter end-to-end learning framework (Li and Marlin, 2016) and training it using an LSTM classifier (Hochreiter and Schmidhuber, 1997). However, training recurrent architectures suffers from the disadvantage that, in practice, they tend to be hard to train to convergence. Only recently, convolutional networks gained much attention in sequence modelling (Gehring *et al.*, 2017; Vaswani *et al.*, 2017), as a viable alternative. For instance, Bai *et al.* (2018) showed that temporal convolutional networks (TCNs), an extension to CNNs employing causal dilated convolutions (Section 3.3.1), outperform conventional recurrent architectures for many sequential learning tasks in terms of evaluation metrics, memory-efficiency, and gradient stability. They concluded that many advantages of TCNs stem from not having to backpropagate in the temporal direction, resulting in easily trainable models of high capacity, despite moderate complexity in terms of parameters. In light of these developments, we propose MGP-TCN, an end-to-end trainable framework for early sepsis detection that builds on both Multi-task Gaussian Process (MGP) Adapters (an extension of Gaussian Process Adapter to multi-task learning) and TCNs. Specifically, MGP-TCN combines the uncertainty-aware framework of GP Adapters with TCNs. The *contributions* of our work are threefold:

- We present DTW-KNN, a multivariate extension of a classic method employing data mining techniques for time series classification, and MGP-TCN, the first method that can leverage temporal convolutions on irregularly sampled multivariate time series.
- Furthermore, we provide the first fully-accessible framework for the early detection of sepsis on a benchmark dataset featuring a publicly available temporally resolved Sepsis-3 label to enable community-based sepsis detection research.
- We present a detailed experimental setup in which we empirically demonstrate that our methods outperform the state of the art in detecting sepsis *early*.

2 Related Work

In this section we introduce the recent literature and current challenges for sepsis detection. We start with the general setup of supervised learning on medical time series, followed by a discussion of existing sepsis detection algorithms. Finally, we conclude with background details about the methods our model builds on.

2.1 Supervised learning on medical time series

Supervised learning on time series datasets has been haunted by the crux that time-wise resolved labels are often missing, especially in medical applications (Reddy and Aggarwal, 2015). This hindrance also applies to the early detection of sepsis. In previous work, it was usually circumvented by applying ad-hoc schemes to determine resolved sepsis labels (Calvert *et al.*, 2016; Mao *et al.*, 2018; Kam and Kim, 2017). These papers used a global time series label (such as an ICD disease code intended for billing) and estimated sepsis onset with an easily computable ad-hoc criterion, such as for example fulfilling the SIRS criteria, which clinicians reconsidered to be unspecific and obsolete (Beesley and Lanspa, 2015; Kaukonen *et al.*, 2015). When using such a patchwork label, it remains unclear whether the patient *actually* suffered from an event at this time—and not, for instance, one week later.

By extracting Sepsis-3, the most recent international sepsis criterion (Singer *et al.*, 2016) which allows for temporal resolution, we contribute to the solution of this issue that affects the Machine Learning for Healthcare literature. Even though there exist datasets with high-resolution sepsis labels, they are currently not *accessible* to the research community, due to either hospitals (Futoma *et al.*, 2017a) or commercially-interested parties (Desautels *et al.*, 2016) preventing data access. Thus, there is a massive threshold to overcome before novel approaches for sepsis detection can be developed.

2.2 Early Sepsis Detection Algorithms

Overview In the last decade, several data-driven approaches for detecting sepsis in the ICU have been presented (Desautels *et al.*, 2016; Calvert *et al.*, 2016; Kam and Kim, 2017; Futoma *et al.*, 2017a; Shashikumar *et al.*, 2017). Many approaches selectively compare to simple clinical scores, such as SIRS (Bone *et al.*, 1992), NEWS (Williams *et al.*, 2012), or MEWS (Stenhouse *et al.*, 2000), none of which are intended as specific, continuously evaluated risk scores for sepsis. Rather than being optimized for performance in terms of sepsis detection, those clinical scores were developed to be easy-to-use for bedside evaluation and they usually track rather broad targets such as generic acute illness or deterioration. Furthermore, Henry *et al.* (2015) introduced a targeted real-time warning score (TREWScore) to predict septic shock, a frequent complication following from sepsis. But notably, only few papers have actually compared to other machine learning approaches. For instance, the application of LSTMs (Kam and Kim, 2017) showed an improvement over the InSight model (Calvert *et al.*, 2016), a regression model with hand-crafted features.

State of the Art Sepsis detection methods are usually developed on real-world datasets with prevalence values ranging from 6.6% (Kam and Kim, 2017) to 21.4% (Futoma *et al.*, 2017a), affecting the difficulty of the dataset-specific task. Despite this considerable class imbalance, to our knowledge, only Futoma *et al.* (2017a) and Desautels *et al.* (2016) report area under the precision–recall curve, i.e. AUPRC, in addition to area under the ROC curve, i.e. AUC. Despite being more commonly used, given the class imbalance, AUC is known to be a less informative evaluation criterion (Saito and Rehmsmeier, 2015). Thus, in terms of AUPRC, Futoma *et al.* (2017a) currently represent the state of the art in early detection of sepsis. In their follow-up paper, Futoma *et al.* (2017b) improved their performance by proposing task-specific tweaks (without code) such as label-propagation, additional feature extraction (e.g. missingness indicators), and separate task correlation matrices for their Gaussian Process. However, those extensions do not modify their LSTM-classifier, but merely the data preprocessing as well as the GP Adapter framework. Hence, those extensions are orthogonal to our undertaking of *improving* the classifier inside the GP Adapter framework.

Limitations of the State of the Art From a methodological point of view, approaches based on deep learning—in particular recurrent architectures for sequence modelling—are attractive for tasks such as sepsis prediction. However, in the recent past, training such architectures suffered from gradient stability issues, i.e. exploding and vanishing gradients. While

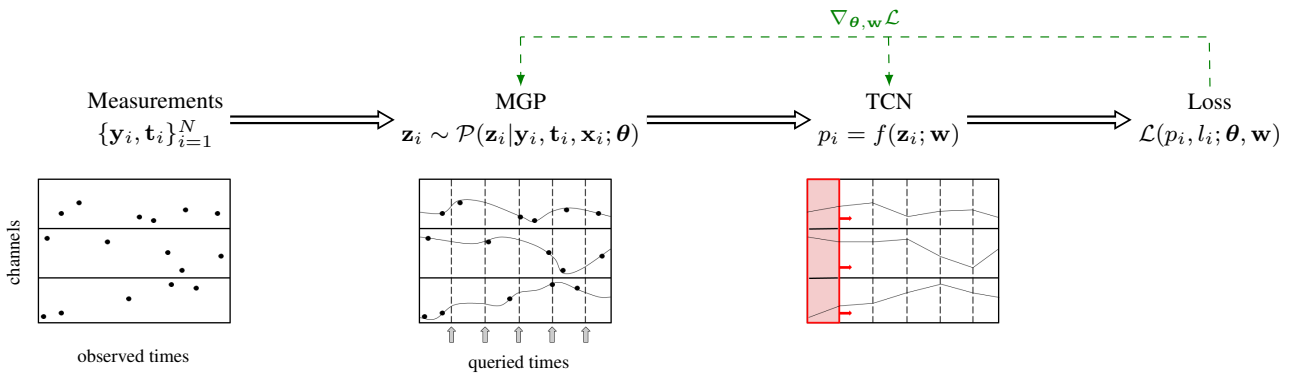

Fig. 1. Overview of our model. The raw, irregularly spaced time series are provided to the Multi-task Gaussian Process (MGP) patient by patient. The MGP then draws from a posterior distribution (given the observed data) at evenly spaced grid times (each hour). This grid is then fed into a temporal convolutional network (TCN) which after a forward pass returns a loss. Its gradient is then computed by backpropagating through the computational graph including both the TCN and the MGP (green arrows). Both the MGP and TCN parameters are learned end-to-end during training.

the last decade saw LSTMs gaining popularity due to addressing this, they still require considerably more memory and can lead to overparametrized models featuring ad-hoc parts that are hard to justify (Jozefowicz *et al.*, 2015). Moreover, understanding the inner workings of RNNs is still challenging because recurrent connections lead to entanglement, which makes it hard to ascertain the function of one neuron without considering the others (Karpathy *et al.*, 2015).

2.3 Temporal Convolutional Networks

Throughout the last decade, Convolutional Neural Networks (CNNs) have gained much popularity, foremost in visual tasks (Cireşan *et al.*, 2011, 2012). CNNs have also shown remarkable successes in other domains such as, for example, text understanding and machine translation (Zhang and LeCun, 2015; Dauphin *et al.*, 2016). Many of them belong to the realm of sequence modelling, which is believed to be "dominated" by recurrent networks (Goodfellow *et al.*, 2016): for instance, Bai *et al.* (2018) put several canonical recurrent architectures to an empirical test on a large battery of datasets and found that temporal convolutional networks (TCN), an extension of conventional CNNs (Lea *et al.*, 2017), outperformed recurrent architectures in most set-ups.

2.4 Gaussian Process Adapters

Li and Marlin (2016) showed that optimizing a Gaussian Process imputation of a time series end-to-end using the gradients of a subsequent classifier leads to better performance than optimizing both the classifier and the GP separately. This method, also referred to as GP Adapters, is not restricted to imputing missing data. For instance, it has also been used to derive latent EEG-electrodes to remove abundant feature dimensions (Li *et al.*, 2017). Recently, Futoma *et al.* (2017a) demonstrated that GP Adapters are a well-suited framework to handle irregularly spaced time series in early sepsis detection. Specifically, they confirmed earlier findings (Li and Marlin, 2016) that in time series classification, GP Adapters outperform conventional GP imputation schemes requiring a separate optimization step which is not driven by the classification task.

3 Methods

In the following, we describe our proposed MGP-TCN and DTW-KNN methods. As for notation, we use regular font for scalars, bold lower-case for vectors and bold upper-case for matrices. First, Section 3.1 gives a high-level overview of our model. The subsequent sections then introduce its individual building blocks: Section 3.2 outlines the MGP Adapter module of our method, which as a whole was previously applied by Futoma *et al.* (2017a), whereas MGPs were originally introduced by Bonilla *et al.* (2008) and the GP Adapter framework by Li and Marlin (2016). Section 3.3 then describes Temporal Convolutional Networks (TCNs) (Bai *et al.*, 2018), the second pillar of our methods. Finally, Section 3.4 briefly describes our DTW-KNN classifier.

3.1 MGP-TCN: Overview

We frame the early detection of sepsis in the ICU as a multivariate time series classification task. Specifically, we focus on the task of identifying sepsis onset in irregularly-sampled multivariate time series of physiological measurements in ICU patients. Our proposed model uses a Multi-task Gaussian Process (MGP) (Bonilla *et al.*, 2008) that is intrinsically capable of dealing with non-uniform sampling frequencies. More precisely, given irregularly-observed data $\{\mathbf{y}_i, \mathbf{t}_i\}$ of encounter i draws an evenly-spaced time series $\mathbf{z}_i$ following the posterior distribution $\mathcal{P}(\mathbf{z}_i|\mathbf{y}_i, \mathbf{t}_i, \mathbf{x}_i; \boldsymbol{\theta})$ of the MGP (see Equation 6 for more details). This latent time series $\mathbf{z}_i$ then serves as the input to a Temporal Convolutional Network (TCN, Section 3.3) (Bai *et al.*, 2018) that predicts the sepsis label. Making use of the Gaussian Process Adapter framework (Li and Marlin, 2016) permits optimizing this entire process end-to-end with respect to the final classification objective, i.e. identifying sepsis. Figure 1 gives an overview of the MGP-TCN model.

3.2 Multi-task Gaussian Process Adapters

3.2.1 Multi-task Gaussian Processes

We first describe how our approach models individual time series with potentially different sampling frequencies and missing observations. To this end, we use a *Gaussian Process* (GP). GPs are a popular choice to model time series because they can handle variable spacing between observations. In addition, they capture the predictive *uncertainty* associated with missing data. To account for multivariate time series, we make use of Multi-task Gaussian Processes (MGP) (Bonilla *et al.*, 2008) with the tasks representing the different medical variables.

Given a patient encounter i fully-observed at T_i times, we "unroll" the different channels of the time series, gathering the values of D variables in $\mathbf{y}_i = (y_{11}, \ldots, y_{T_i 1}, \ldots, y_{12}, \ldots, y_{T_i 2}, \ldots, y_{1D}, \ldots, y_{T_i D})^T$, and collect all T_i observation times in a vector $\mathbf{t}_i$. In clinical practice, this array is sparse and hence inefficient to store explicitly; we only use it

here for notational convenience. Each encounter receives a binary label l_i indicating whether the patient develops sepsis during this stay.

We model the true value of encounter i and variable d at time t using a latent function $f_{i,d}(t)$. The MGP places a Gaussian Process prior over the latent functions to directly induce the correlation between tasks using a shared correlation function $k^\tau(\cdot,\cdot)$ over time. Assuming zero-meaned GPs, we have:

$$\langle f_{i,d}(t), f_{i,d'}(t')\rangle = K^D_{d,d'}\ k^\tau(t,t') \tag{1}$$

$$y_{i,d}(t) \sim \mathcal{N}\left(f_{i,d}\left(t\right), \sigma_d^2\right), \tag{2}$$

where $\mathbf{K}^D$ is the task-similarity kernel matrix whose entry $K^D_{d,d'}$ at position (d, d') represents the similarity of tasks d and d', while σ_d^2 denotes the noise variance of task d. An entire, fully-observed multivariate time series of encounter i follows

$$\mathbf{y}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_i) \tag{3}$$

$$\mathbf{\Sigma}_i = \mathbf{K}^D \otimes \mathbf{K}^{T_i} + \mathbf{D} \otimes \mathbf{I}, \tag{4}$$

where $\otimes$ denotes the Kronecker product, and $\mathbf{K}^{T_i}$ represents an encounter-specific $T_i \times T_i$ correlation matrix between all observed times $t_i \in \mathbf{t}_i$ of encounter i, while $\mathbf{D}$ is a diagonal matrix of per-task noise variances satisfying $D_{dd} = \sigma_d^2$. Building on previous work on modelling noisy physiological time series (Williams and Rasmussen, 2006; Futoma *et al.*, 2017a), we use an Ornstein–Uhlenbeck kernel as a correlation function, i.e. $k^\tau(t,t';l) := \exp(-|t-t'|/l)$, parametrized using a length scale l. For simplicity, we share $\mathbf{K}^D$ and $k^\tau(\cdot,\cdot;l)$ and the per-task noise variances across different patients. Hence, the parameterization of the MGP can be summarized as

$$\boldsymbol{\theta} = \{\mathbf{K}^D, \sigma_d^2|_{d=1}^D, l\} \tag{5}$$

or D^2+D+1 parameters. In a fully-observed setting, $\mathbf{\Sigma}_i$ is a $D\cdot T_i \times D\cdot T_i$ covariance matrix. However, in clinical practice, only a subset of all D variables is measured at most observation times. Thus, we only have to compute entries of $\mathbf{\Sigma}_i$ for *observed* pairs of time and variable type. So for encounter i, if the number of all observed measurements $m_i < D \cdot T_i$, we only compute an $m_i \times m_i$ covariance matrix.

Following Futoma *et al.* (2017a), we use the MGP to preprocess the sparse and irregularly spaced multi-channel time series of a patient's measurements in order to output an regularly-spaced time series driven by the final classification task. To achieve this, let $\mathcal{X}$ be a list of regularly-spaced points in time that starts with the ICU admission as hour zero and continues by counting the time since admission (in our case) in full hours. Using this grid, for each encounter we derive a vector $\mathbf{x}_i = (x_1, \ldots, x_{X_i})$ of grid times which will be used as query points for the MGP. More specifically, $x_1 = 0$ and $x_{n+1} - x_n = 1$ for all encounters. We use the next full hour after the last observed point in time as the encounter-specific last grid time x_{X_i} (for more details on how we select the patient time window, please refer to Section 4.3). On a patient level, the MGP induces a posterior distribution over the $D \times X_i$ matrix $\mathbf{Z}_i$ of imputed time series values at the X_i queried points in time for D tasks. As previously shown (Bonilla *et al.*, 2008; Futoma *et al.*, 2017a), when stacking the columns of $\mathbf{Z}_i$ such that $\mathbf{z}_i = \text{vec}(\mathbf{Z}_i)$, the posterior distribution follows a multivariate normal distribution

$$\mathbf{z}_i \sim \mathcal{N}\big(\boldsymbol{\mu}(\mathbf{z}_i), \mathbf{\Sigma}(\mathbf{z}_i); \boldsymbol{\theta}\big) \tag{6}$$

with

$$\boldsymbol{\mu}(\mathbf{z}_i) = (\mathbf{K}^D \otimes \mathbf{K}^{X_i T_i})\mathbf{\Sigma}_i^{-1}\mathbf{y}_i \tag{7}$$

$$\begin{aligned}\mathbf{\Sigma}(\mathbf{z}_i) = &(\mathbf{K}^D \otimes \mathbf{K}^{X_i}) \\ &- (\mathbf{K}^D \otimes \mathbf{K}^{X_i T_i})\mathbf{\Sigma}_i^{-1}(\mathbf{K}^D \otimes \mathbf{K}^{T_i X_i}).\end{aligned} \tag{8}$$

Here, $\mathbf{K}^{X_i T_i}$ refers to the correlation matrix between the queried grid times $\mathbf{x}_i$ and the observed times $\mathbf{t}_i$ while $\mathbf{K}^{X_i}$ represents the correlations between $\mathbf{x}_i$ with itself.

3.2.2 Classification Task

So far, we have outlined how the MGP returns a evenly-spaced multi-channel time series $\mathbf{Z}_i$ when given a patient's raw time series data $\{\mathbf{y}_i, \mathbf{t}_i\}$. To train a model and ultimately perform classification, we require a loss function. As Li and Marlin (2016) stated first, if $\mathbf{Z}_i$ were directly observed, it could be directly fed into a off-the-shelf classifier such that its loss could be simply computed as $\mathcal{L}(f(\mathbf{Z}_i;\mathbf{w}), l_i)$ with l_i denoting the class label. However, $\mathbf{Z}_i$ is actually a random variable and so is the loss function. We account for this by using the expectation $\mathbb{E}_{\mathbf{z}_i\sim\mathcal{N}(\boldsymbol{\mu}(\mathbf{z}_i),\mathbf{\Sigma}(\mathbf{z}_i);\boldsymbol{\theta})}[\mathcal{L}(f(\mathbf{Z}_i;\mathbf{w}), l_i)]$ as the overall loss function for optimization. The learning task then becomes minimizing this loss over the entire dataset. Thus, we search parameters $\mathbf{w}^*, \boldsymbol{\theta}^*$ that satisfy:

$$\mathbf{w}^*, \boldsymbol{\theta}^* = \arg\min_{\mathbf{w},\boldsymbol{\theta}} \sum_{i=1}^{N} \overbrace{\mathbb{E}_{\mathbf{z}_i\sim\mathcal{N}(\boldsymbol{\mu}(\mathbf{z}_i),\mathbf{\Sigma}(\mathbf{z}_i);\boldsymbol{\theta})}\big[\mathcal{L}(f(\mathbf{Z}_i;\mathbf{w}), l_i)\big]}^{E_i} \tag{9}$$

For many choices of $f(\cdot)$ the expectation E_i in Equation 9 is analytically not tractable. We thus use Monte Carlo sampling with s samples to approximate this term as

$$E_i \approx \frac{1}{S}\sum_{s=1}^{S} \mathcal{L}(f(\mathbf{Z}_s;\mathbf{w}), l_i), \tag{10}$$

where

$$\text{vec}(\mathbf{Z}_s) = \mathbf{z}_s \sim \mathcal{N}\big(\boldsymbol{\mu}(\mathbf{z}_i), \mathbf{\Sigma}(\mathbf{z}_i); \boldsymbol{\theta}\big). \tag{11}$$

To compute the gradients of this expression with respect to both the classifier parameters $\mathbf{w}$ and the MGP parameters $\boldsymbol{\theta}$, we make use of a reparametrization (Kingma and Welling, 2014) and set $\mathbf{z}_i = \boldsymbol{\mu}(\mathbf{z}_i) + \mathbf{R}\boldsymbol{\xi}$, where $\mathbf{R}$ satisfies $\mathbf{\Sigma}(\mathbf{z}_i) = \mathbf{R}\mathbf{R}^T$ and $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. In this work, for the sake of simplicity, we use a Cholesky decomposition to determine $\mathbf{R}$ and refrain from more involved approximative techniques (such as the Lanczos approach used by Li and Marlin (2016), for example).

3.3 Temporal Convolutional Networks

In this section, we outline the details of a generic temporal convolutional network architecture. This architecture is advantageous for sequence

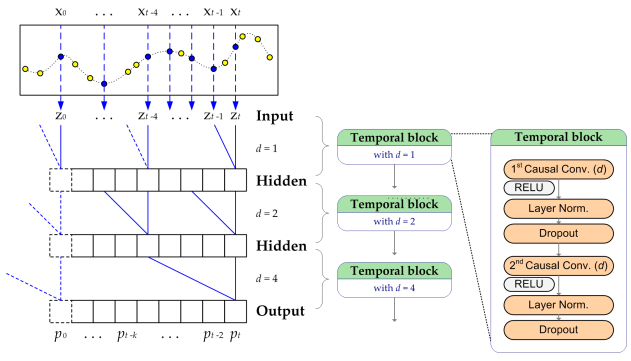

Fig. 2. Schematic of the TCN architecture. The input $\mathbf{z}_i$ values (blue) to the TCN classifier are computed by the Multi-Task Gaussian process on a regular grid $x_0, \ldots, x_t$ based on the observed values (yellow). Each temporal block skips an increasing number of the previous layer's outputs, such that the visible window of a single neuron increases exponentially with increasing number of layers. Figure recreated from Bai et al. (2018).

modeling because it is less susceptible to gradient stability issues. Figure 2 illustrates our TCN architecture, while the subsequent section provides more details.

3.3.1 Causal Dilated Convolutions

TCNs can be seen as a simple but powerful extension to conventional 1D-CNNs in that they show three properties (Bai *et al.*, 2018):

1. Given an input sequence, a TCN returns an output of same length (sequence to sequence).
2. An output value can only be influenced by present and past input values (causal convolutions).
3. Assuming constant filter sizes, when going deeper into the architecture, the receptive field, and thus also the effective memory, grows exponentially (dilated convolutions).

For 1. and 2., we use zero-padding to enforce equal layer sizes throughout all layers while making sure that the convolutions are "tilted" such that for the output at time t only input values at t and earlier can be used. For 3., we follow the work of Yu and Koltun (2015) by defining the l-dilated convolution of an input sequence $\mathbf{x}$ with a filter $\mathbf{f}$ as

$$(\mathbf{x} *_l \mathbf{f})(k) = \sum_{k=i+l\cdot j} x_i \cdot f_j, \tag{12}$$

where the 1-dilated convolution coincides with the regular convolution. By stacking l-dilated convolutions in an exponential manner, such that $l = 2^n$ for the n-th layer, a long effective memory can be achieved as illustrated by Bai *et al.* (2018).

3.3.2 Residual Temporal Blocks

Here we describe how convolutional layers of a TCN are stacked into residual temporal blocks, i.e. blocks that combine the previous input and the result of the respective convolution with an addition. Thus, the output of a temporal block is computed relatively with respect to an input. The only functional difference to the residual blocks of Bai *et al.* (2018) is that for training stability, Lee (2018) and our method both apply layer normalization (Ba *et al.*, 2016) instead of weight normalization (Salimans and Kingma, 2016). Both techniques were recently derived from batch normalization (Ioffe and Szegedy, 2015) to speed up convergence without mini-batch dependencies. This is achieved by normalizing the layers inputs or the layers weights, respectively. As an additional detail, the code by Lee (2018) applies normalization after the activation step, whereas in Bai *et al.* (2018) it occurs before activation. Whether to normalize before or after activation is an ongoing debate. For instance, Mishkin *et al.* (2017) showed that normalizing after activation empirically can improve performance. We follow the setup of Lee (2018).

3.4 Dynamic Time Warping Classifier

Dynamic time warping (Keogh and Pazzani, 1999) for time series classification, using k-nearest neighbor approaches, here referred to as DTW-KNN, is known to exhibit highly competitive predictive performance (Dau *et al.*, 2017; Ding *et al.*, 2008). As opposed to many other off-the-shelf classifiers, it can handle variable-length time series. Surprisingly, despite its wide-spread use and demonstrated capabilities in data mining, DTW-KNN has (to the best of our knowledge) never been used in sepsis detection tasks. We thus extend DTW-KNN for the classification of multivariate time series, thereby introducing an *additional* novel approach for sepsis early detection. More precisely, we address the multivariate nature of our setup by computing the DTW distance matrix (containing the pairwise distances between all patients) for each time series channel separately. Each distance matrix is subsequently used for training a k-nearest neighbor (KNN) classifier. Instead of using resampling techniques, the ensemble is constructed by combining *all* per-channel classifiers. Finally, for the classification step, the final prediction score is computed as the average over all per-channel prediction scores.

4 Experiments

4.1 Dataset

In our analysis, we use the MIMIC-III (*Multiparameter Intelligent Monitoring in Intensive Care*) database, version 1.4 (Johnson *et al.*, 2016). MIMIC-III includes over 58,000 hospital admissions of over 45,000 patients, as encountered between June 2001 and October 2012. In Table 1 we summarize patient statistics.

We follow the most recent sepsis definition (Singer *et al.*, 2016) which requires a co-occurrence of suspected infection (SI) and organ dysfunction. For SI, we follow the recommendations of Seymour *et al.* (2016) for implementing the SI cohort (for more details, see Supplementary Section A.1.2). According to Singer *et al.* (2016), organ dysfunction is fulfilled when the Sequential Organ Failure Assessment (SOFA) Score (Vincent *et al.*, 1996) shows an acute increase of ≥ 2 points. For determining the organ dysfunction criterion, we follow the suggestions of the authors to use a specified window of $-48\,\mathrm{h}$ to $24\,\mathrm{h}$ around suspicion of infection (Singer *et al.*, 2016). The Sepsis-3 implementation is visualized in Figure 3. For detecting sepsis *early*, determining the sepsis onset time is crucial. We thus considerably refined the queries provided by Johnson *et al.* (2018) to determine the Sepsis-3 label on an hourly basis, as their provided code only checks whether a simplified version of Sepsis-3 is satisfied upon admission (for instance, no *increase* in SOFA points is considered, but only one abnormally high value). If sepsis is determined by only checking whether a patient fulfills the criteria upon admission, only those patients who *arrive* in the ICU with sepsis would be defined as cases, not the—arguably more interesting ones—that *develop* the syndrome during their ICU stay.

4.2 Data Filtering

Patient Inclusion Criteria We exclude patients under the age of 15 and those for which no chart data is available—including ICU admission or discharge time. Furthermore, following the recent sepsis literature, ICU encounters logged via the CareVue system were excluded due to underreported negative lab measurements (Desautels *et al.*, 2016). We include an encounter as a case if at any time during the ICU stay a sepsis onset occurs, whereas controls are defined as those patients that have no sepsis onset (they still might have suspected infection or organ dysfunction, separately). To additionally ensure that controls cannot be sepsis patients that developed sepsis shortly before ICU, we require controls not to be labeled with any sepsis-related ICD-9 billing code. Following the above inclusion criteria, we initially count 1,797 sepsis cases and 17,276 controls.

Table 1. Summary statistics of the Sepsis-3 patient cohort including controls. The mean sepsis onset is given in hours since admission to the ICU.

Variable	**Sepsis Cases**	**Controls**
n	1,797	17,276
Female	742 (41.3%)	7,678 (44.4%)
Male	1,055 (58.7%)	9,598 (55.6%)
Age ($\mu \pm \sigma$)	67.6 ± 15.1	63.9 ± 17.4
Sepsis Onset Time	$7.9\,\mathrm{h}$	–
Admission Type		
Emergency	1,497	14,521
Elective	274	2,566
Urgent	26	189

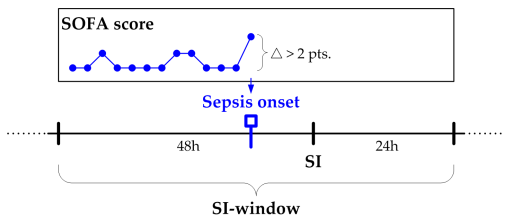

Fig. 3. Following (Singer et al., 2016; Seymour et al., 2016), for each encounter with a suspicion of infection (SI), we extract a 72 h window around the first SI event (starting 48 h before) as the SI-Window. The SOFA score (Vincent et al., 1996) is then evaluated for every hour of this window, following the SOFA definition in considering the worst values of the last 24 h.

This works aims for sepsis *early* detection, so we follow Desautels *et al.* (2016) and exclude cases that develop sepsis earlier than seven hours into the ICU stay (this enables a seven hour prediction horizon). To preserve a realistic class balance of around 10%, we apply this exclusion step only after the case-control matching (see below). Thus, after cleaning and filtering we finally use 570 cases and 5,618 controls for analysis. As for variables, we used 44 irregularly sampled laboratory and vital parameters (for more details, see Section A.2). Furthermore, for being able to run all baselines we had add to apply an additional patient filtering step (for details, refer to Section A.7).

Case–Control Matching In previous work, it has been observed that an insufficient alignment of time series of sepsis cases versus controls could render the classification task trivial. For instance, when comparing a window before sepsis onset to the last window before discharge) of the controls ICU stay, the classification task is much easier than when comparing to a more similar reference time in the controls stay. This can be observed by the decrease in performance of the MGP-RNN method when Futoma *et al.* applied case-control matching in their follow-up work (Futoma *et al.*, 2017b). Hence, to avoid a trivial classification task, we also use a case-control alignment in a matching procedure. To accommodate the class imbalance, we assign each case to 10 random unassigned controls and define their control onset as the absolute time (in hours since admission) when the matched case fulfilled the sepsis criteria. Futoma *et al.* (2017b) used a relative measure, i.e. the same percentage of the entire ICU stay as for control onset time. However, we observed that cases and controls do not necessarily share the same length of stay, which could introduce bias to the alignment that a deep neural network could potentially exploit.

4.3 Experimental Setup

In this section we, outline the experimental setup, with respect to baselines, training and evaluation.

Baselines We compare our method against MGP-RNN, the current state of the art sepsis detection method (Futoma *et al.*, 2017a,b). To enable a fair comparison, the authors kindly provided source code for their pipeline such that their model could be trained from scratch on our dataset. Additionally, we compare against a classical TCN (here referred to as "raw" TCN) that is not embedded in the MGP Adapter framework. To this end, as a preprocessing step, we first impute the times series using a carry-forward scheme (for more details, please refer to Section A.3). Using the same imputation scheme, we train our DTW-KNN ensemble.

Training We apply a 3-fold cross-validation with random splits using 80% of the samples for training and each 10% for validation and testing. After each random split, the time series were z-scored per channel using the respective train mean and standard deviation. Due to costly evaluations, instead of an exhaustive grid search, for hyperparameter tuning we apply an off-the-shelf Bayesian optimization as provided by the `scikit-optimize` library (the scikit-optimize contributers, 2018) with 20 calls per method and split. We select the best model parameters and checkpoints in terms of validation AUPRC and evaluate them on the test splits. For all deep models, the hyperparameter spaces were constrained such that the number of parameters each ranged from 20K–500K. To make a cross-validation setting feasible (despite a deep learning setup), we constrain each run to take at most two hours (for more details, see Section A.6). To guard against underfitting, we additionally retrained the best parameter setting of each method and split for a longer period of 50 epochs (please refer to Section A.6).

Evaluation Due to substantial class imbalance (the overall case prevalence is 9.2%, or roughly 1 case for 10 controls), we evaluate all models in terms of area under the precision–recall curve (AUPRC) on the test split. In addition, we report area under the receiver–operator curve (AUC), mostly to comply with recent sepsis detection literature (please see Section 2.2 for a discussion of the disadvantages of this measure). Since timely identification of sepsis is of central importance, we evaluate the trained models in a horizon analysis going back up to 7 hours before sepsis onset. For example, to evaluate the prediction horizon at 3 hours in advance, for each encounter the model is only provided input data up until this moment. Note that to assess the predictive performance we do not optimize the models to each respective horizon hour (resulting in 8 distinct specialized models). Instead, per method (and fold), we train one model on all available training data and challenge its performance by gradually restricting access to the information closest to the sepsis onset event.

4.4 Implementation Details & Runtimes

For maximal reproducibility, we embedded our method and all comparison partners in the `sacred v0.7.4` environment (Klaus Greff *et al.*, 2017). A local installation of the MIMIC-III database was done with PostgreSQL 9.3.22. The queries to extract the data from the database are based on queries in the public `mimic-code` repository (Johnson *et al.*, 2018). However, to extract the hourly-resolved sepsis label, we had to implement an entire query pipeline on top of the original code (Johnson *et al.*, 2018). For the MGP module, we included the code implemented by (Futoma *et al.*, 2017a) with minor changes. For the TCNs, we extend the TensorFlow implementation of Lee (2018) We further apply *gradient checkpointing* (Chen *et al.*, 2016) for all neural network models in order to permit training in a typical GPU setup. The DTW-KNN classifier was implemented using the libraries `tslearn v0.1.26` (Tavenard, 2017) for dynamic time warping and `scikit-learn v0.20.2` (Pedregosa *et al.*, 2011) for the k-nearest neighbor classifier. We implemented both our proposed methods as well as all comparison partners in Python 3. All experiments were performed on a Ubuntu 14.04.5 LTS server with 2 CPUs (Intel® Xeon® E5-2620 v4 @ 2.10GHz), 8 GPUs (NVIDIA® GeForce® GTX 1080), and 128 GiB of RAM. However, for the deep learning models, we exclusively used single GPU processing. Table 2 depicts the runtimes of all methods.

Table 2. Training runtimes (for the RNN/TCN methods, this includes the sum of all three splits, whereas for DTW-KNN distances were computed only once).

Method	RAW-TCN	MGP-RNN	MGP-TCN	DTW-KNN
Runtime	49.7 h	74.8 h	73.4 h	136.9 h

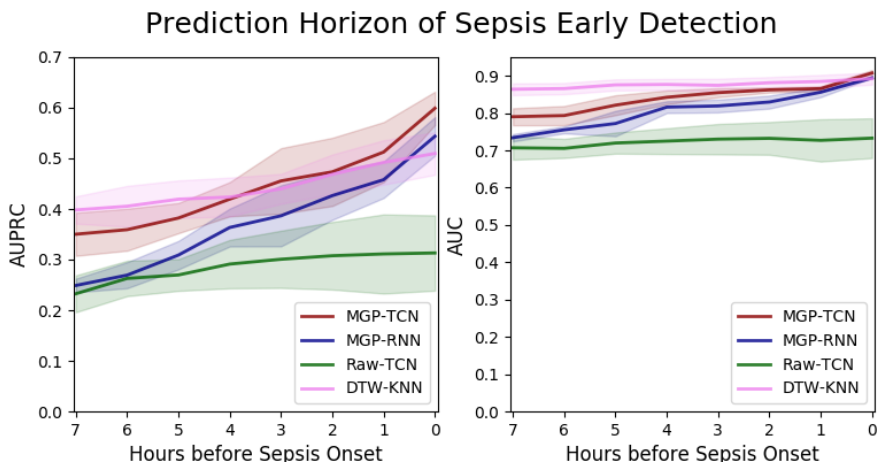

Fig. 4. We evaluate all methods using Area under the Precision–Recall Curve (AUPRC) and additionally display the (less informative) Area under the Receiver Operator Characteristic (AUC). The current state-of-the-art method, MGP-RNN, is shown in blue. The two approaches for early detection of sepsis that were introduced in this paper, i.e. MGP-TCN and DTW-KNN ensemble, are shown in pink and red, respectively. By using three random splits for all measures and methods, we depict the mean (line) and standard deviation error bars (shaded area).

4.5 Results

This section presents our experimental results. Each method was optimized on the full train data split on each of three folds. To challenge the models, we evaluated the predictive performance by providing gradually less input information when *evaluating* (for details, refer to the last paragraph of Section 4.3). Figure 4 displays our findings. On the x-axes the prediction horizon is indicated in hours before sepsis onset. On the y-axes, we measure AUPRC (left) and AUC (right). Due to the substantial class imbalance (9.2%), we focus our evaluation on AUPRC.

We observe that both our novel model MGP-TCN and our DTW-KNN ensemble method *consistently* outperform the current state-of-the-art early sepsis detection classifier MGP-RNN. Especially for the early detection task of more than 4 hours before sepsis onset, both MGP-TCN and DTW-KNN outperform the state of the art with a significant margin, while the latter shows slightly higher performance in this regime. Nevertheless, it should be noted that *all* approaches except Raw-TCN exhibit overlapping performance in terms of their variance estimates for $0\,\mathrm{h}$ to $3\,\mathrm{h}$ prior to onset.

Interestingly, the Raw-TCN approach does not yield competitive results for *any* setting that was considered in our experimental setup. With increasing distance to the onset, its performance starts to approximate that of the MGP-RNN classifier.

5 Discussion

Our proposed methods MGP-TCN and DTW-KNN exhibit favorable performances over all prediction horizons and consistently outperform the state-of-the-art baseline classifier MGP-RNN. Comparing to the classic TCN, we empirically demonstrated that TCN-based architectures, in combination with Multi-Task Gaussian Process Adapters (MGP), result in competitive predictive performance. Specifically, in terms of AUPRC, with MGP-TCN and DTW-KNN, we improve the current state of the art from 0.25 to 0.35 and 0.40 seven hours before sepsis onset. Furthermore, this also confirms that recent advances in sequence modeling may be transferred successfully to irregularly-sampled scenarios. By contrast, the low performance of the Raw-TCN classifier shows that the imputation scheme is crucial for transferring "deep" approaches to our scenario. When comparing our findings with the recent literature, we observe that the low prevalence in our dataset ($9.2\,\%$) makes the classification task substantially harder. For instance, in terms of AUPRC, MGP-RNN performed better on the original dataset (to which we unfortunately have no access) that exhibits a prevalence of $21.4\,\%$ (Futoma *et al.*, 2017a). In terms of prevalence and preprocessing, to our knowledge the "least incomparable" setup would be the one by Desautels *et al.* (2016); however, we were denied access to their method and their queries. Interestingly, their AUPRC dropped to roughly 0.3 already at one hour before sepsis onset, where for example, our proposed MGP-TCN method still achieves an AUPRC of 0.51.

One of the most surprising results is the highly-competitive performance of our (conceptually simple) DTW-KNN ensemble classifier. In particular, its performance for earlier horizons exceeds all other methods. While this is highly relevant for sepsis early detection, the DTW-KNN classifier suffers from some practical limitations which impede online monitoring scenarios. First, it unfortunately does not exhibit favorable scaling behavior, as already for our data set, training each horizon may require the alignment of up to 1.5 *billion* univariate time series, followed by their pairwise distance computation, and the subsequent storing of results (which potentially affects both runtime and memory). We conjecture that DTW-KNN performs so well due to the mid-range sample size (whereas deep models usually perform best for even larger sample sizes). However, scaling DTW-KNN to cohorts of hundreds of thousands of patients currently appears to be computationally infeasible. This also affects online classification in the ICU: for each new measurement of a patient, the distances of each involved channel to *all* patients in the training cohort have to be updated, and partially recomputed. As a consequence, intermediate results of the distance calculations need to be stored, leading to a significant memory overhead. The problem thus remains a "hot topic" in time series analysis (Oregi *et al.*, 2017).

By contrast, MGP-TCN does not suffer from these limitations, as only the network weights have to be stored, and prediction is *constant* in the number of patients. Thus, MGP-TCN can be easily trained on larger cohorts, which is likely to further increase predictive performance. Moreover, obtaining online predictions only requires very slight modifications. For more details on the scaling behavior, please refer to Section A.5.

For future work, we aim to extend our analysis to more types of data sources arising from the ICU. Futoma *et al.* (2017b) already employed a subset of baseline covariates, medication effects, and missingness indicator variables. However, a multitude of feature classes still remain to be explored and properly integrated. For instance, the combination

of sequential and non-sequential features has previously been handled by feeding non-sequential data into the sequential model (Futoma *et al.*, 2017a). We hypothesize that this could be handled more efficiently by using a more modular architecture that incorporates both sequential and non-sequential parts. Furthermore, we aim to obtain a better understanding of the time series features utilized by the model. Specifically, we are interested in assessing the *interpretability* of the learned filters of the MGP-TCN framework and evaluate how much the activity of an individual filter contributes to a prediction. This endeavor is somewhat facilitated by our use of a convolutional architecture. The extraction of short per-channel signals could prove very relevant for supporting diagnoses made by clinical practitioners.

In conclusion, we proposed a novel model (MGP-TCN) that, to our knowledge, constitutes the first setup that successfully transfers recent advances in convolutional sequence modeling to irregularly-spaced time series. In addition, we showed that a classic data mining approach, when extended to multivariate time series classification, can outperform the current state of the art in the early detection of sepsis. We empirically demonstrated that our approaches are capable of detecting sepsis earlier and more accurately than competitor approaches and thus have the potential to help clinical practitioners manage sepsis both effectively and in a timely manner.

Funding

This work was funded in part by the SPHN/PHRT Driver Project "Personalized Swiss Sepsis Study"

References

Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.

Bai, S., Kolter, J. Z., and Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Beesley, S. J. and Lanspa, M. J. (2015). Why we need a new definition of sepsis. *Annals of translational medicine*, **3**(19).

Bone, R. C., Balk, R. A., Cerra, F. B., Dellinger, R. P., Fein, A. M., Knaus, W. A., Schein, R. M., and Sibbald, W. J. (1992). Definitions for sepsis and organ failure and guidelines for the use of innovative therapies in sepsis. *Chest*, **101**(6), 1644–1655.

Bonilla, E. V., Chai, K. M., and Williams, C. (2008). Multi-task gaussian process prediction. In *Advances in Neural Information Processing Systems*, pages 153–160.

Calvert, J. S., Price, D. A., Chettipally, U. K., Barton, C. W., Feldman, M. D., Hoffman, J. L., Jay, M., and Das, R. (2016). A computational approach to early sepsis detection. *Computers in Biology and Medicine*, **74**, 69–73.

Chen, T., Xu, B., Zhang, C., and Guestrin, C. (2016). Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*.

Cireşan, D., Meier, U., and Schmidhuber, J. (2012). Multi-column deep neural networks for image classification. *arXiv preprint arXiv:1202.2745*.

Cireşan, D. C., Meier, U., Masci, J., Maria Gambardella, L., and Schmidhuber, J. (2011). Flexible, high performance convolutional neural networks for image classification. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1237–1242.

Dau, H. A., Silva, D. F., Petitjean, F., Forestier, G., Bagnall, A., and Keogh, E. (2017). Judicious setting of Dynamic Time Warping's window width allows more accurate classification of time series. In *Proceedings of the IEEE International Conference on Big Data*, pages 917–922.

Dauphin, Y. N., Fan, A., Auli, M., and Grangier, D. (2016). Language modeling with gated convolutional networks. *arXiv preprint arXiv:1612.08083*.

Dellinger, R. P., Levy, M. M., Rhodes, A., Annane, D., Gerlach, H., Opal, S. M., Sevransky, J. E., Sprung, C. L., Douglas, I. S., Jaeschke, R., Osborn, T. M., Nunnally, M. E., Townsend, S. R., Reinhart, K., Kleinpell, R. M., Angus, D. C., Deutschman, C. S., Machado, F. R., Rubenfeld, G. D., Webb, S. A., Beale, R. J., Vincent, J.-L., and Moreno, R. (2013). Surviving sepsis campaign: International guidelines for management of severe sepsis and septic shock 2012. *Critical Care Medicine*, **41**(2), 580–637.

Desautels, T., Calvert, J., Hoffman, J., Jay, M., Kerem, Y., Shieh, L., Shimabukuro, D., Chettipally, U., Feldman, M. D., Barton, C., *et al.* (2016). Prediction of sepsis in the intensive care unit with minimal electronic health record data: A machine learning approach. *JMIR Medical Informatics*, **4**(3), e28.

Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., and Keogh, E. (2008). Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, **1**(2), 1542–1552.

Ferrer, R., Martin-Loeches, I., Phillips, G., Osborn, T. M., Townsend, S., Dellinger, R. P., Artigas, A., Schorr, C., and Levy, M. M. (2014). Empiric antibiotic treatment reduces mortality in severe sepsis and septic shock from the first hour: results from a guideline-based performance improvement program. *Critical Care Medicine*, **42**(8), 1749–1755.

Futoma, J., Hariharan, S., and Heller, K. (2017a). Learning to detect sepsis with a multitask Gaussian process RNN classifier. In *International Conference on Machine Learning (ICML)*, pages 1174–1182.

Futoma, J., Hariharan, S., Sendak, M., Brajer, N., Clement, M., Bedoya, A., O'Brien, C., and Heller, K. (2017b). An improved multi-output Gaussian process RNN with real-time validation for early sepsis detection. *arXiv preprint arXiv:1708.05894*.

Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N. (2017). Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*.

Golub, G. H. and Van Loan, C. F. (2012). *Matrix computations*, volume 3. JHU Press.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*. MIT Press, Cambridge, MA, USA.

Henry, K. E., Hager, D. N., Pronovost, P. J., and Saria, S. (2015). A targeted real-time early warning score (TREWScore) for septic shock. *Science Translational Medicine*, **7**(299).

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, **9**(8), 1735–1780.

Hotchkiss, R. S., Moldawer, L. L., Opal, S. M., Reinhart, K., Turnbull, I. R., and Vincent, J. L. (2016). Sepsis and septic shock. *Nature Reviews Disease Primers*, **2**.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Johnson, A. E., Stone, D. J., Celi, L. A., and Pollard, T. J. (2018). The MIMIC Code Repository: enabling reproducibility in critical care research. *Journal of the American Medical Informatics Association*, **25**(1), 32–39.

Johnson, A. E. W., Pollard, T. J., Shen, L., Lehman, L.-w. H., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Anthony Celi, L., and Mark, R. G. (2016). MIMIC-III, a freely accessible critical care database. *Scientific Data*, **3**.

Jozefowicz, R., Zaremba, W., and Sutskever, I. (2015). An empirical exploration of recurrent network architectures. In *International Conference on Machine Learning*, pages 2342–2350.

Kam, H. J. and Kim, H. Y. (2017). Learning representations for the early detection of sepsis with deep neural networks. *Computers in biology and medicine*, **89**, 248–255.

Karpathy, A., Johnson, J., and Fei-Fei, L. (2015). Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*.

Kaukonen, K. M., Bailey, M., Suzuki, S., Pilcher, D., and Bellomo, R. (2014). Mortality related to severe sepsis and septic shock among critically ill patients in Australia and New Zealand, 2000-2012. *Journal of the American Medical Association (JAMA)*, **311**(13), 1308–1316.

Kaukonen, K.-M., Bailey, M., Pilcher, D., Cooper, D. J., and Bellomo, R. (2015). Systemic inflammatory response syndrome criteria in defining severe sepsis. *New England Journal of Medicine*, **372**(17), 1629–1638.

Keogh, E. J. and Pazzani, M. J. (1999). Scaling up dynamic time warping to massive datasets. In J. M. Żytkow and J. Rauch, editors, *Principles of Data Mining and Knowledge Discovery*, pages 1–11, Heidelberg, Germany. Springer.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*.

Klaus Greff, Aaron Klein, Martin Chovanec, Frank Hutter, and Jürgen Schmidhuber (2017). The Sacred Infrastructure for Computational Research. In Katy Huff, David Lippa, Dillon Niederhut, and M. Pacer, editors, *Proceedings of the 16th Python in Science Conference*, pages 49 – 56.

Lea, C., Flynn, M. D., Vidal, R., Reiter, A., and Hager, G. D. (2017). Temporal convolutional networks for action segmentation and detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1003–1012. IEEE.

Lee, C. (2018). Solving sequential MNIST with Temporal Convolutional Networks (TCNs).

Li, S. C.-X. and Marlin, B. M. (2016). A scalable end-to-end Gaussian process adapter for irregularly sampled time series classification. In *Advances in Neural Information Processing Systems*, pages 1804–1812.

Li, Y., Dzirasa, K., Carin, L., Carlson, D. E., *et al.* (2017). Targeting EEG/LFP synchrony with neural nets. In *Advances in Neural Information Processing Systems*, pages 4620–4630.

Mao, Q., Jay, M., Hoffman, J. L., Calvert, J., Barton, C., Shimabukuro, D., Shieh, L., Chettipally, U., Fletcher, G., Kerem, Y., *et al.* (2018). Multicentre validation of a sepsis prediction algorithm using only vital sign data in the emergency department, general ward and icu. *BMJ open*, **8**(1), e017833.

Mishkin, D., Sergievskiy, N., and Matas, J. (2017). Systematic evaluation of convolution neural network advances on the imagenet. *Computer Vision and Image Understanding*.

Oregi, I., Pérez, A., Del Ser, J., and Lozano, J. A. (2017). On-line dynamic time warping for streaming time series. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 591–605. Springer.

Peake, S., Bellomo, R., Cameron, P., Cross, A., Delaney, A., Finfer, S., George, C., Higgins, A., Jones, D., Moran, J., Myburgh, J., Syres, G., Webb, S., and Williams, P. (2007). The outcome of patients with sepsis and septic shock presenting to emergency departments in Australia and New Zealand. *Critical Care and Resuscitation*, **9**(1), 8–18.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, **12**, 2825–2830.

Reddy, C. K. and Aggarwal, C. C. (2015). *Healthcare data analytics*. Chapman and Hall/CRC.

Saito, T. and Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, **10**(3), e0118432.

Salimans, T. and Kingma, D. P. (2016). Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems*, pages 901–909.

Seymour, C. W., Liu, V. X., Iwashyna, T. J., Brunkhorst, F. M., Rea, T. D., Scherag, A., Rubenfeld, G., Kahn, J. M., Shankar-Hari, M., Singer, M., *et al.* (2016). Assessment of clinical criteria for sepsis: For the Third International Consensus Definitions for Sepsis and Septic Shock (Sepsis-3). *Journal of the American Medical Association (JAMA)*, **315**(8), 762–774.

Shashikumar, S. P., Stanley, M. D., Sadiq, I., Li, Q., Holder, A., Clifford, G. D., and Nemati, S. (2017). Early sepsis detection in critical care patients using multiscale blood pressure and heart rate dynamics. *Journal of Electrocardiology*.

Singer, M., Deutschman, C. S., Seymour, C. W., Shankar-Hari, M., Annane, D., Bauer, M., Bellomo, R., Bernard, G. R., Chiche, J.-D., Coopersmith, C. M., *et al.* (2016). The Third International Consensus Definitions for Sepsis and Septic Shock (Sepsis-3). *Journal of the American Medical Association (JAMA)*, **315**(8), 801–810.

Stenhouse, C., Coates, S., Tivey, M., Allsop, P., and Parker, T. (2000). Prospective evaluation of a modified early warning score to aid earlier detection of patients developing critical illness on a general surgical ward. *British Journal of Anaesthesia*, **84**(5), 663P.

Tavenard, R. (2017). tslearn: A machine learning toolkit dedicated to time-series data. `https://github.com/rtavenar/tslearn`.

the scikit-optimize contributers (2018). scikit-optimize/scikit-optimize: v0.5.2. `https://doi.org/10.5281/zenodo.1207017`.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Vincent, J.-L., Moreno, R., Takala, J., Willatts, S., De Mendonça, A., Bruining, H., Reinhart, C., Suter, P., and Thijs, L. (1996). The SOFA (sepsis-related organ failure assessment) score to describe organ dysfunction/failure. *Intensive Care Medicine*, **22**(7), 707–710.

Williams, B., Alberti, G., Ball, C., Bell, D., Binks, R., Durham, L., *et al.* (2012). National early warning score (news): Standardising the assessment of acute-illness severity in the nhs. *London: The Royal College of Physicians*.

Williams, C. K. and Rasmussen, C. E. (2006). *Gaussian processes for machine learning*. MIT Press, Cambridge, MA, USA.

Yu, F. and Koltun, V. (2015). Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*.

Zhang, X. and LeCun, Y. (2015). Text understanding from scratch. *arXiv preprint arXiv:1502.01710*.

A Supplementary Material

A.1 Sepsis-3 Implementation

Following Singer *et al.* (2016) and Seymour *et al.* (2016), for each encounter with a suspicion of infection (SI), for the first SI event we extract the 72 hours window around it (starting 48 hours before) as the SI-Window.

A.1.1 Suspicion of Infection

To determine suspicion of infection, we follow Seymour *et al.* (2016)'s definition of the suspected infection (SI) cohort. The SI criterion manifests in the timely co-occurrence of antibiotic administration and body fluid sampling. If a culture sample was obtained before the antibiotic, the drug had to be ordered within 72 hours. If the antibiotic was administered first the sampling had to follow within 24 hours.

A.1.2 Organ Dysfunction

The SOFA score (Vincent *et al.*, 1996) is a scoring system which Sepsis-3 suggests to use for assessing organ dysfunction. Since this is a particularly time-sensitive matter, we evaluate the SOFA score (which considers the worst parameters of the previous 24 hours) at each hour of the 72 hour window around suspicion of infection. More importantly, as Sepsis-3 foresees it, to ensure an acute increase in SOFA of ≥ 2 points, we trigger the organ dysfunction criterion when SOFA has *increased* by 2 points during this window.

A.2 List of Clinical Variables

Table A.1 lists all used clinical variables, comprising 44 vital and laboratory parameters.

Table A.1. List of all 44 used clinical variables. For this study, we focused on irregularly sampled time series data comprising vital and laboratory parameters. To ensure availability, we only selected variables that surpassed a number of 500 non-NaNs in the entire dataset.

Vital Parameters	
Systolic Blood Pressure	Tidal Volume set
Diastolic Blood Pressure	Tidal Volume observed
Mean Blood Pressure	Tidal Volume spontaneous
Respiratory Rate	Peak Inspiratory Pressure
Heart Rate	Total Peep Level
SpO2 (Pulsoxymetry)	O2 flow
Temperature Celsius	FiO2 (fraction of inspired oxygen)
Cardiac Output	
Laboratory Parameters	
Albumin	Blood Urea Nitrogen
Bands (immature neutrophils)	White blood cells
Bicarbonate	Creatine Kinase
Bilirubin	Creatine Kinase MB
Creatinine	Fibrinogen
Chloride	Lactate Dehydrogenase
Sodium	Magnesium
Potassium	Calcium (free)
Lactate	pO2 Bloodgas
Hematocrit	pH Bloodgas
Hemoglobin	pCO2 Bloodgas
Platelet count	SO2 bloodgas
Partial Thromboplastin Time	Glucose
Prothrombin Time (Quick)	Troponin T
INR (standardized Quick)	

A.3 Imputation Schemes

Here we provide more details about how the methods that do *not* employ an MGP were imputed. For maximal comparability to the MGP sampling frequency, we binned the time series into bins of one hour width by taking the mean of all measurements inside this window. Then, we apply a carry-forward imputation scheme were empty bins are filled with the value of the last non-empty one. The remaining empty bins (at the start of the time series) were then mean-imputed (which after centering was reduced to 0 imputation).

A.4 Hyperparameter Search

Differentiable models For the differentiable models *MGP-RNN*, *MGP-TCN*, and *RAW-TCN* an extensive hyperparameter search based on Bayesian optimization was performed using the `scikit-optimize` package (the scikit-optimize contributers, 2018). More precisely, we relied on a Gaussian Process to model the AUPRC of the models dependent on the hyperparameters. The models were then trained at the hyperparameter values that matched one of the randomly-selected criteria *largest confidence bounds*, *largest expected improvement* and *highest probability of improvement* according to the Gaussian Process prior. Ten initial evaluations were performed at random according to the hyperparameter search space, followed by ten evaluations according to the Gaussian Process prior. During the hyperparameter search phase, the *MGP-RNN* model was trained for 5 epochs over the complete dataset—since we observed fast convergence behavior—while the TCN based model were trained 15 and 100 epochs for the *MGP-TCN* and the *RAW-TCN* model respectively.

DTW-KNN classifier The performance of the DTW-KNN classifier was evaluated on the same validation dataset as the other models for $k \in \{1, 3, \ldots, 13, 15\}$, while relying on the training dataset with 0 hours before Sepsis onset. The k value yielding the best AUPRC on the validation dataset was selected for subsequent evaluation on the testing dataset. Similar to the other classifiers, we do not "refit" the KNN classifier by removing data from the training dataset to generate the horizon plots.

A.5 Additional Information on Scaling

Li and Marlin (2016) demonstrated that the MGP Adapter framework is dominated by drawing from the MGP distribution. Thus, in our case, inverting $\mathbf{\Sigma_i} \in \mathbb{R}^{D \cdot T_i \times D \cdot T_i}$ has a computational complexity of $\mathcal{O}(D^3 \cdot T_i^3)$ (Golub and Van Loan, 2012). Notably, classifying a patient *only* depends on the length of the current patient time series and the number of tasks/variables; it does *not* depend on the number of patients in the training data set, i.e. it has a complexity of $\mathcal{O}(1)$ in the number of patients.

By contrast, while a naive implementation of DTW-KNN has a very low training complexity ($\mathcal{O}(1)$, due to its character as a "lazy learner"), the complexity at prediction time is very high. In order to classify a single instance, the KNN classifier requires the distances of the instance to all N training points. Moreover, the runtime of a single distance computation using Dynamic Time Warping (DTW) is $\mathcal{O}(DT^2)$, where D represents the number of channels in the time series and T is the length of the shorter time series. Overall, this leads to a runtime complexity of $\mathcal{O}(NDT^2)$ for a *single* prediction step, which can quickly become infeasible for large-sized heath record datasets, especially if online predictions are desired. Consequently, already for $N \geq D^2T$, the complexity of the DTW-KNN approach will exceed that of the MGP-TCN. Furthermore, the cubic complexity in the prediction step can be ameliorated by using faster approximation schemes (Li and Marlin, 2016).

Table A.2. Detailed information about hyperparameter search ranges. *: we fixed 10 monte carlo samples according to Futoma et al. (2017a). **: the MGP-RNN baseline was presented with a batch-size of 100, thus we did not enforce our range on this baseline (Futoma et al., 2017a).

Model	**Hyperparameter**	**Lower bound**	**Upper bound**	**Sampling distribution**
All models	learning rate	5e−4	5e−3	log uniform
	Monte Carlo Samples	10*		not applicable
MGP-TCN RAW-TCN	batch size	10	40	uniform
	temporal blocks	4	9	uniform
	filters per layer	15	90	uniform
	filter width	2	5	uniform
	dropout	0	0.1	uniform
	L_2-regularization	0.01	100	log uniform
MGP-RNN	batch size	100**		not applicable
	layers	1	3	uniform
	hidden units per layer	20	150	uniform
	L_2-regularization	1e−4	1e−3	log uniform

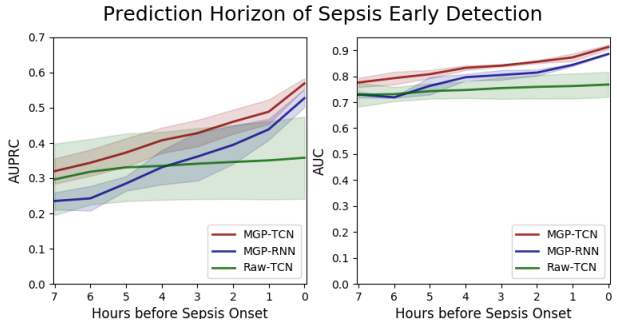

Fig. A.1. The auxiliary setup as computed for the full horizon. We retrained the best parameter settings of all deep model for a prolonged time to guard against underfitting. Due to a slightly decreased performance, the MGP-based models show a tendency to overfit when training for 50 epochs, whereas the RAW-TCN shows high variance between the random splits.

A.6 Supplementary Results

To enforce a maximum time of two hours per call for each method (in order to make different hyperparameter searches on several folds feasible), MGP-RNN trains for 5 epochs, MGP-TCN for 15, and Raw-TCN for 100. We applied early stopping based on validation AUPRC with patience $= 5$ epochs.

Moreover, in an auxiliary setup, to guard against underfitting, we use the best parameter setting of each deep model and retrain each model for a prolonged period (50 epochs for both MGP-based models, 100 epochs for RAW-TCN) using early stopping based on validation AUPRC with patience $= 10$ epochs. As shown in Figure A.6, the deep models exhibit a similar tendency as in Figure 4, with the exception of the RAW-TCN showing high variability (due to one fold with favorable performance). Also, the MGP-based models exhibit a slight drop in performance. This may indicate that in terms of epochs, they converge earlier and overfit earlier.

A.7 Additional Information for the Horizon Analysis

When creating the horizon analysis, we observed that the current MGP-RNN implementation (using a Lanczos iteration) requires the minimal number of observed measurements of a patient to be at least the number of Monte Carlo samples (i.e. 10 in our case). Hence, we performed an on-the-fly masking of encounters that did *not* satisfy this criterion; for comparability, we applied it to all models. Table A.3 details the patient counts obtained after masking. Additionally, to be able fit all methods into memory, we removed a single outlier encounter consisting of more than 10K measurements. Notably, as we did not refit the models on each horizon (as this would answer a different question), with increasing prediction horizon, the slight decrease in sample size should not bias the mean performance but rather scale up the error bars.

Table A.3. Patients count after applying masking which was required for making the MGP-RNN baseline work.

Horizon	**Split**	Train			Validation			Test		
	Fold	0	1	2	0	1	2	0	1	2
0		4953	4950	4950	618	618	619	617	620	619
1		4951	4947	4947	618	618	619	616	620	619
2		4943	4941	4937	616	617	619	616	617	619
3		4933	4933	4927	614	615	617	615	614	618
4		4913	4917	4910	613	611	616	613	611	613
5		4832	4827	4830	602	605	602	598	600	600
6		4587	4580	4565	566	570	581	567	570	574
7		4073	4061	4056	503	508	510	497	504	507