Pattern Recognition 115 (2021) 107896

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Time series cluster kernels to exploit informative missingness and incomplete label information

Karl Øyvind Mikalsen[a,b,1,*], Cristina Soguero-Ruiz[c], Filippo Maria Bianchi[d], Arthur Revhaug[b], Robert Jenssen[a,1]

[a] Department of Physics and Technology, UiT The Arctic University of Norway, Tromsø, Norway
[b] Department of Gastrointestinal Surgery, University Hospital of North Norway (UNN), Tromsø, Norway
[c] Department of Signal Theory and Comm., Telematics and Computing, Universidad Rey Juan Carlos, Fuenlabrada, Spain
[d] Department of Mathematics and Statistics, UiT, Tromsø, Norway

ARTICLE INFO

Article history:
Received 27 November 2018
Revised 11 November 2020
Accepted 8 February 2021
Available online 20 February 2021

Keywords:
Multivariate time series
Kernel methods
Missing data
Informative missingness
Semi-supervised learning

ABSTRACT

The time series cluster kernel (TCK) provides a powerful tool for analysing multivariate time series subject to missing data. TCK is designed using an ensemble learning approach in which Bayesian mixture models form the base models. Because of the Bayesian approach, TCK can naturally deal with missing values without resorting to imputation and the ensemble strategy ensures robustness to hyperparameters, making it particularly well suited for unsupervised learning.

However, TCK assumes missing at random and that the underlying missingness mechanism is ignorable, i.e. uninformative, an assumption that does not hold in many real-world applications, such as e.g. medicine. To overcome this limitation, we present a kernel capable of exploiting the potentially rich information in the missing values and patterns, as well as the information from the observed data. In our approach, we create a representation of the missing pattern, which is incorporated into mixed mode mixture models in such a way that the information provided by the missing patterns is effectively exploited. Moreover, we also propose a semi-supervised kernel, capable of taking advantage of incomplete label information to learn more accurate similarities.

Experiments on benchmark data, as well as a real-world case study of patients described by longitudinal electronic health record data who potentially suffer from hospital-acquired infections, demonstrate the effectiveness of the proposed methods.

1. Introduction

Multivariate time series (MTS) frequently occur in a whole range of practical applications such as medicine, biology, and climate studies, to name a few. A challenge that complicates the analysis is that real-world MTS are often subject to large amounts of missing data. Traditionally, missingness mechanisms have been categorized into missing completely at random (MCAR), missing at random (MAR) and missing not at random (MNAR) [1]. The main difference between these mechanisms consists in whether the missingness is ignorable (MCAR and MAR) or non-ignorable (MNAR) [1–3]. In e.g. medicine, non-ignorable missingness can occur when the missing patterns R are related to the disease under study Y. In this case, the distribution of the missing patterns for diseased patients is not equal to the corresponding distribution for the control group, i.e. $p(R \mid Y = 1) \neq p(R \mid Y = 0)$. Hence, the missingness is *informative* [4–7]. By contrast, uninformative missingness will be referred to as *ignorable* in the remainder of this paper.

Both ignorable and informative missingness occur in real-world data. An example from medicine of ignorable missingness occurs e.g. if a clinician orders lab tests for a patient and the tests are performed, but because of an error the results are not recorded. On the other hand, informative missingness could occur if it is decided to not perform lab tests because the doctor thinks the patient is in good shape. In the latter case, the missing values and patterns potentially contain rich information about the diseases and clinical outcomes for the patient. Efficient data-driven approaches aiming to extract knowledge, perform predictive modelling, etc., must be capable of capturing this information.

* Corresponding author.
E-mail address: karl.o.mikalsen@uit.no (K. Øyvind Mikalsen).
[1] KM, RJ are with the UiT Machine Learning Group: machine-learning.uit.no.

https://doi.org/10.1016/j.patcog.2021.107896

Various methods have been proposed to handle missing data in MTS [8–11]. One simple approach is to create a *complete* dataset by discarding the time series with missing data. However, this gives unbiased predictions only if the missingness mechanism is MCAR. As an alternative, a preprocessing step involving *imputation* of missing values with some estimated value, such as the mean, is common. Other so-called *single imputation* methods exploit machine learning based methods such as multilayer perceptrons, self-organizing maps, k-nearest neighbors, recurrent neural networks and regression-based imputation [12,13]. Alternatively, one can impute missing values using various smoothing and interpolation techniques [12,14]. Among these, a prominent example is the last observation carried forward (LOCF) scheme that imputes the last non-missing value for the following missing values. Limitations of imputation methods are that they introduce additional bias and they ignore uncertainty associated with the missing values.

Multiple imputation [15] resolves this problem, to some extent, by estimating the missing values multiple times and thereby creating multiple complete datasets. Thereafter, e.g. a classifier is trained on all datasets and the results are combined to obtain the final predictions. However, despite that multiple imputation and other imputation methods can give satisfying results in some scenarios, these are ad-hoc solutions that lead to a multi-step procedure in which the missing data are handled separately and independently from the rest of the analysis. Moreover, the information about which values are actually missing (the missing patterns) is lost, i.e. imputation methods cannot exploit informative missingness.

Due to the aforementioned limitations, several research efforts have been devoted over the last years to process incomplete time series without relying on imputation [5–7,16–23]. In this regard, powerful kernel methods have been proposed, of which the *time series cluster kernel* (TCK) [24] is a prominent example. The TCK is designed using an ensemble learning approach in which Bayesian mixture models form the base models. An advantage of TCK, compared to imputation methods, is that the missing data are handled automatically and no additional tasks are left to the user. Multiple imputation instead requires a careful selection of the imputation model and other variables are needed to do the imputation [8], which particularly in an unsupervised setting can turn out to be problematic.

A shortcoming of the TCK is that unbiased predictions are only guaranteed for ignorable missingness, i.e. the kernel cannot take advantage of informative missing patterns frequently occurring in medical applications. To overcome this limitation, in this work, we present a novel time series cluster kernel, TCK_{IM}. In our approach, we create a representation of the missing patterns using masking, i.e. we represent the missing patterns using binary indicator time series. By doing so, we obtain MTS consisting of both continuous and discrete attributes. To model these time series, we introduce mixed mode Bayesian mixture models, which can effectively exploit information provided by the missing patterns.

The time series cluster kernels are particularly useful in unsupervised settings. In many practical applications such as e.g. medicine it is not feasible to obtain completely labeled training sets [25], but in some cases it is possible to annotate a few samples with labels, i.e. incomplete label information is available. In order to exploit the incomplete label information, we propose a semi-supervised MTS kernel, ssTCK. In our approach, we incorporate ideas from information theory to measure similarities between distributions. More specifically, we employ the Kullback-Leibler divergence to assign labels to unlabeled data.

Experiments on benchmark MTS datasets and a real-world case study of patients suffering from hospital-acquired infections, described by longitudinal electronic health record data, demonstrate the effectiveness of the proposed TCK_{IM} and ssTCK kernels.

The remainder of this paper is organized as follows. Section 2 presents background on MTS kernels. The two proposed kernels are described in Sections 3 and 4, respectively. Experiments on synthetic and benchmark datasets are presented in Section 5, whereas the case study is described in Section 6. Section 7 concludes the paper.

2. Multivariate time series kernels to handle missing data

Kernel methods have been of great importance in machine learning for several decades and have applications in many different fields [26–28]. Within the context of time series, a *kernel* is a similarity measure that also is positive semi-definite [29,30]. Once defined, such similarities between pairs of time series may be utilized in a wide range of applications, such as classification or clustering, benefiting from the vast body of work in the field of kernel methods. Here we provide an overview of MTS kernels, and describe how they deal with missing data.

The simplest of all kernel functions is the linear kernel, which for two data points represented as vectors, x and y, is given by the inner product $\langle x, y \rangle$, possibly plus a constant c. One can also apply a linear kernel to pairs of MTS once they are unfolded into vectors. However, by doing so the information that they are MTS and there might be inherent dependencies in time and between attributes, is then lost. Nevertheless, in some cases such a kernel can be efficient, especially if the MTS are short [31]. If the MTS contain missing data, the linear kernel requires a preprocessing step involving e.g. imputation.

The most widely used time series similarity measure is *dynamic time warping* (DTW) [32–34], where the similarity is quantified as the alignment cost between the MTS. More specifically, in DTW the time dimension of one or both of the time series is warped to achieve a better alignment. Despite the success of DTW in many applications, similarly to many other similarity measures, it is non-metric and therefore cannot non-trivially be used to design a positive semi-definite kernel [35]. Hence, it is not suited for kernel methods in its original formulation. However, because of its popularity there have been attempts to design kernels exploiting the DTW. For example, Cuturi et al. designed a DTW-based kernel using global alignments [36]. An efficient version of the global alignment kernel (GAK) is provided in Cuturi [37]. The latter has two hyperparameters, namely the kernel bandwidth and the triangular parameter. GAK does not naturally deal with missing data and incomplete datasets, and therefore also requires a preprocessing step involving imputation.

Two MTS kernels that can naturally deal with missing data without having to resort to imputation are the *learned pattern similarity* (LPS) [38] and TCK. LPS generalizes the well-known autoregressive models to local autopatterns using multiple lag values for autocorrelation. These autopatterns are supposed to capture the local dependency structure in the time series and are learned using a tree-based (random forest) learning strategy. More specifically, a time series is represented as a matrix of segments. Randomness is injected to the learning process by randomly choosing time segment (column in the matrix) and lag p for each tree in the random forest. A bag-of-words type compressed representation is created from the output of the leaf-nodes for each tree. The final time series representation is created by concatenating the representation obtained from the individual trees, which in turn are used to compute the similarity using a histogram intersection kernel [39].

The TCK is based on an ensemble learning approach wherein robustness to hyperparameters is ensured by joining the clustering results of many Gaussian mixture models (GMM) to form the

final kernel. Hence, no critical hyperparameters have to be tuned by the user, and the TCK can be learned in an unsupervised manner. To ensure robustness to sparsely sampled data, the GMMs that are the base models in the ensemble, are extended using informative prior distributions such that the missing data is explicitly dealt with. More specifically, the TCK matrix is built by fitting GMMs to the set of MTS for a range of number of mixture components. The idea is that by generating partitions at different resolutions, one can capture both the local and global structure of the data. Moreover, to capture diversity in the data, randomness is injected by for each resolution (number of components) estimating the mixture parameters for a range of random initializations and randomly chosen hyperparameters. In addition, each GMM sees a random subset of attributes and segments in the MTS. The posterior distributions for each mixture component are then used to build the TCK matrix by taking the inner product between all pairs of posterior distributions. Eventually, given an ensemble of GMMs, the TCK is created in an additive way by using the fact that the sum of kernels is also a kernel. Recently, TCK has also been extended to handle spatial dependencies [40].

Despite that LPS and TCK kernels share many properties, the way missing data are dealt with is very different. In LPS, the missing data handling abilities of decision trees are exploited. Along with ensemble methods, fuzzy approaches and support vector solutions, decision trees can be categorized as *machine learning approaches for handling missing data* [12], i.e. the missing data are handled naturally by the machine learning algorithm. One can also argue that the way missing data are dealt with in the TCK belongs to this category, since an ensemble approach is exploited. However, it can also be categorized as a *likelihood-based approach* since the underlying models in the ensemble are Gaussian mixture models. In the likelihood-based approaches, the full, incomplete dataset is analysed using maximum likelihood (or maximum a posteriori, equivalently), typically in combination with the expectation-maximization (EM) algorithm [8,9]. These approaches assume that the missingness is ignorable.

3. Time series cluster kernel to exploit informative missingness

In this section, we present the novel time series cluster kernel, TCK_{IM}, which is capable of exploiting informative missingness.

A key component in the time series cluster kernel framework is ensemble learning, in which the basic idea consists in combining a collection of many base models into a composite model. A good such composite model will have statistical, computational and representational advantages such as lower variance, lower sensitivity to local optima and is capable of representing a broader span functions (increased expressiveness), respectively, compared to the individual base models [41]. Key to achieve this is *diversity* and *accuracy* [42], i.e. the base models cannot make the same errors on new test data and have to perform better than random guessing. This can be done by integrating multiple outcomes of the same (weak) base model as it is trained under different, often randomly chosen, settings (parameters, initialization, subsampling, etc.) to ensure diversity [43].

In the TCK_{IM} kernel, the base model is a mixed mode Bayesian mixture model. Next, we provide the details of this model.

Notation

The following notation is used. A multivariate time series (MTS) X is defined as a (finite) combination of univariate time series (UTS), $X = \{x_v \in \mathbb{R}^T \mid v = 1, 2, \ldots, V\}$, where each attribute, x_v, is a UTS of length T. The number of UTS, V, is the *dimension* of X. The length T of the UTS x_v is also the length of the MTS X. Hence, a V-dimensional MTS, X, of length T can be represented as a matrix in $\mathbb{R}^{V\times T}$. Given a dataset of N MTS, we denote $X^{(n)}$ the n-th MTS. An incompletely observed MTS is described by the pair $U^{(n)} = (X^{(n)}, R^{(n)})$, where $R^{(n)}$ is a binary MTS with entry $r_v^{(n)}(t) = 0$ if the realization $x_v^{(n)}(t)$ is missing and $r_v^{(n)}(t) = 1$ if it is observed.

Mixed mode mixture model

Assume that a MTS $U = (X, R)$ is generated from two modes. X is a V-variate real-valued MTS ($X \in \mathbb{R}^{V\times T}$), whereas R is a V-variate binary MTS ($R \in \{0, 1\}^{V\times T}$). Further, we assume that U is generated from a finite mixture density,

$$p(U \mid \Phi, \Theta) = \sum_{g=1}^{G} \theta_g f(U \mid \phi_g), \tag{1}$$

where G is the number of components, f is the density of the components parametrized by $\Phi = (\phi_1, \ldots, \phi_G)$, and $\Theta = (\theta_1, \ldots, \theta_g)$ are the mixing coefficients, $0 \le \theta_G \le 1$ and $\sum_{g=1}^{G} \theta_g = 1$.

Now, introduce a latent random variable Z, represented as a G-dimensional one-hot vector $Z = (Z_1, \ldots, Z_G)$, whose marginal distribution is given by $p(Z \mid \Theta) = \prod_{g=1}^{G} \theta_g^{Z_g}$. The unobserved variable Z records the membership of U and therefore $Z_g = 1$ if U belongs to component g and $Z_g = 0$ otherwise. Hence, $p(U \mid Z, \Phi) = \prod_{g=1}^{G} f(U \mid \phi_g)^{Z_g}$, and therefore it follows that

$$p(U, Z \mid \Phi, \Theta) = p(U \mid Z, \Phi) p(Z \mid \Theta) = \prod_{g=1}^{G} [f(U \mid \phi_g)\theta_g]^{Z_g} \tag{2}$$

$U = (X, R)$ consists of two modalities X and R. We now naively assume that

$$f(U \mid \phi_g) = f(X \mid R, \mu_g, \Sigma_g) f(R \mid \beta_g), \tag{3}$$

where $f(X \mid R, \mu_g, \Sigma_g)$ is a density function given by

$$f(X \mid R, \mu_g, \Sigma_g) = \prod_{v=1}^{V} \prod_{t=1}^{T} \mathcal{N}(x_v(t) \mid \mu_{gv}(t), \sigma_{gv})^{r_v(t)}, \tag{4}$$

and $f(R \mid \beta_g)$ is a probability mass given by

$$f(R \mid \beta_g) = \prod_{v=1}^{V} \prod_{t=1}^{T} \beta_{gvt}^{r_v(t)} (1 - \beta_{gvt})^{1 - r_v(t)}. \tag{5}$$

The parameters of each component are $\phi_g = (\mu_g, \Sigma_g, \beta_g)$, where $\mu_g = \{\mu_{gv} \in \mathbb{R}^T \mid v = 1, \ldots, V\}$ is a time-dependent mean (μ_{gv} is a UTS of length T), $\Sigma_g = diag\{\sigma_{g1}^2, \ldots, \sigma_{gV}^2\}$ is a time-constant diagonal covariance matrix in which σ_{gv}^2 is the variance of attribute v, and $\beta_{gvt} \in [0, 1]$ are the parameters of the Bernoulli mixture model in Eq. (5). The idea is that even though the missingness mechanism is ignored in $f(X \mid R, \mu_g, \Sigma_g)$, which is only computed over the observed data, the Bernoulli term $f(R \mid \beta_g)$ will capture information from the missing patterns.

The conditional probability of Z given U, can be found using Bayes' theorem,

$$\pi_g \equiv P(Z_g = 1 \mid U, \Phi, \Theta) = \frac{\theta_g \prod_{v=1}^{V} \prod_{t=1}^{T} [\mathcal{N}(x_v(t) \mid \mu_{gv}(t), \sigma_{gv})\beta_{gvt}]^{r_v(t)} (1 - \beta_{gvt})^{1 - r_v(t)}}{\sum_{g=1}^{G} \theta_g \prod_{v=1}^{V} \prod_{t=1}^{T} [\mathcal{N}(x_v(t) \mid \mu_{gv}(t), \sigma_{gv})\beta_{gvt}]^{r_v(t)} (1 - \beta_{gvt})^{1 - r_v(t)}}. \tag{6}$$

Similarly to [24], we introduce a Bayesian extension and put informative priors over the parameters of the normal distribution, which enforces smoothness over time and that clusters containing

few time series, to have parameters similar to the mean and covariance computed over the whole dataset. A kernel-based Gaussian prior is defined for the mean, $P(\mu_{gv}) = \mathcal{N}\left(\mu_{gv} \mid m_v,\ S_v\right).m_v$ are the empirical means and the prior covariance matrices, S_v, are defined as $S_v = s_v \mathcal{K}$, where s_v are empirical standard deviations and $\mathcal{K}$ is a kernel matrix, whose elements are $\mathcal{K}_{tt'} = b_0 \exp(-a_0(t - t')^2)$, $t, t' = 1, \ldots, T$. a_0, b_0 are user-defined hyperparameters. An inverse Gamma distribution prior is put on the standard deviation σ_{gv}, $P(\sigma_{gv}) \propto \sigma_{gv}^{-N_0} \exp\left(-\frac{N_0 s_v}{2\sigma_{gv}^2}\right)$, where N_0 is a user-defined hyperparameter. We denote $\Omega = \{a_0, b_0, N_0\}$ the set of hyperparameters.

Then, given a dataset $\{U^{(n)}\}_{n=1}^N$, the parameters $\{\Phi, \Theta\}$ can be estimated using maximum a posteriori expectation maximization (MAP-EM) [44,45]. This leads to Algorithm 1.

Algorithm 1 MAP-EM for mixed mode mixture model.

Require: Dataset $\{U^{(n)} = (X^{(n)}, R^{(n)})\}_{n=1}^N$, hyperparameters Ω and number of mixtures G.

1: Initialize the parameters $\Theta = (\theta_1, \ldots, \theta_G)$ and $\Phi = \{\mu_g, \sigma_g, \beta_g\}_{g=1}^G$.

2: E-step. For each MTS $U^{(n)}$, evaluate the posterior probabilities using Eq. (6) with the current parameter estimates.

3: M-step. Update parameters using the current posteriors

$$\theta_g = N^{-1} \sum_{n=1}^N \pi_g^{(n)}$$

$$\sigma_{gv}^2 = \frac{N_0 s_v^2 + \sum_{n=1}^N \sum_{t=1}^T r_v^{(n)}(t)\, \pi_g^{(n)} \left(x_v^{(n)}(t) - \mu_{gv}(t)\right)^2}{N_0 + \sum_{n=1}^N \sum_{t=1}^T r_v^{(n)}(t)\, \pi_g^{(n)}}$$

$$\mu_{gv} = \frac{S_v^{-1} m_v + \sigma_{gv}^{-2} \sum_{n=1}^N \pi_g^{(n)} \mathrm{diag}(r_v^{(n)})\, x_v^{(n)}}{S_v^{-1} + \sigma_{gv}^{-2} \sum_{n=1}^N \pi_g^{(n)} \mathrm{diag}(r_v^{(n)})}$$

$$\beta_{gvt} = (\textstyle\sum_{n=1}^N \pi_g^{(n)})^{-1} \sum_{n=1}^N \pi_g^{(n)} r_v^{(n)}(t)$$

4: Repeat step 2-3 until convergence.

Ensure: Posteriors $\Pi^{(n)} \equiv \left(\pi_1^{(n)}, \ldots, \pi_G^{(n)}\right)^T$ and parameter estimates Θ and Φ.

3.1. Forming the kernel

We now explain how the mixed mode mixture model is used to form the TCK_{IM} kernel.

We use the mixed mode Bayesian mixture model as the base model in an ensemble approach. To ensure diversity, we vary the number of components for the base models by sampling from a set of integers $\mathcal{I}_C = \{I, \ldots, I + C\}$. For each number of components, we apply Q different random initial conditions and hyperparameters. We let $\mathcal{Q} = \{q = (q_1, q_2) \mid q_1 = 1, \ldots Q,\ q_2 \in \mathcal{I}_C\}$ be the index set keeping track of initial conditions and hyperparameters (q_1), and the number of components (q_2). Each base model q is trained on a random subset of MTS $\{(X^{(n)}, R^{(n)})\}_{n \in \eta(q)}$. Moreover, for each q, we select random subsets of variables $\mathcal{V}(q)$ as well as random time segments $\mathcal{T}(q)$.

The inner products of the normalized posterior distributions from each mixture component are then added up to build the TCK_{IM} kernel matrix. Note that, in addition to introducing novel base models to account for informative missingness, we also modify the kernel by normalizing the vectors of posteriors to have unit length in the l_2-norm. This provides an additional regularization that may increase the generalization capability of the learned model. The details of the method are presented in Algorithm 2. The kernel for MTS not available during training can be evaluated according to Algorithm 3.

Algorithm 2 Time series cluster kernel. Training phase.

Require: Training set of MTS $\{(X^{(n)}, R^{(n)})\}_{n=1}^N$, Q initializations, set of integers $\mathcal{I}_C$ controlling number of components for each base model.

1: Initialize kernel matrix $K = 0_{N \times N}$.

2: **for** $q \in \mathcal{Q}$ **do**

3: Compute posteriors $\Pi^{(n)}(q) \equiv (\pi_1^{(n)}, \ldots, \pi_{q_2}^{(n)})^T$, by fitting a mixed mode mixture model with q_2 clusters to the dataset and by randomly selecting:

i. hyperparameters $\Omega(q)$,
ii. a time segment $\mathcal{T}(q)$ of length $T_{min} \leq |\mathcal{T}(q)| \leq T_{max}$ to extract from each $X^{(n)}$ and $R^{(n)}$,
iv. a subset of attributes $\mathcal{V}(q)$, with cardinality $V_{min} \leq |\mathcal{V}(q)| \leq V_{max}$, to extract from each $X^{(n)}$ and $R^{(n)}$,
vi. a subset of MTS, $\eta(q)$, with $N_{min} \leq |\eta(q)| \leq N$,
vii. initialization of the mixture parameters $\Theta(q)$ and $\Phi(q)$.

4: Update kernel matrix, $K_{nm} = K_{nm} + \frac{\Pi^{(n)}(q)^T \Pi^{(m)}(q)}{\|\Pi^{(n)}(q)\| \cdot \|\Pi^{(m)}(q)\|}$.

5: **end for**

Ensure: K kernel matrix, time segments $\mathcal{T}(q)$, subsets of attributes $\mathcal{V}(q)$, subsets of MTS $\eta(q)$, parameters $\Theta(q)$, $\Phi(q)$ and posteriors $\Pi^{(n)}(q)$.

Algorithm 3 Time series cluster kernel. Test phase.

Require: Test set $\left\{X^{*(m)}\right\}_{m=1}^M$, time segments $\mathcal{T}(q)$ subsets of attributes $\mathcal{V}(q)$, $\mathcal{V}_R(q)$, subsets of MTS $\eta(q)$, parameters $\Theta(q)$, $\Phi(q)$ and posteriors $\Pi^{(n)}(q)$.

1: Initialize kernel matrix $K^* = 0_{N \times M}$.

2: **for** $q \in \mathcal{Q}$ **do**

3: Compute posteriors $\Pi^{*(m)}(q)$, $m = 1, \ldots, M$ using the mixture parameters $\Theta(q)$, $\Phi(q)$.

4: Update kernel matrix, $K^*_{nm} = K^*_{nm} + \frac{\Pi^{(n)}(q)^T \Pi^{*(m)}(q)}{\|\Pi^{(n)}(q)\| \cdot \|\Pi^{*(m)}(q)\|}$.

5: **end for**

Ensure: K^* test kernel matrix.

4. Semi-supervised time series cluster kernel

This section presents a semi-supervised MTS kernel, ssTCK, capable of exploiting incomplete label information. In ssTCK, the base mixture models are learned exactly in the same way as in TCK or TCK_{IM}, i.e. if there is no missing data, or the missingness is ignorable, the base models will be the Bayesian GMMs. Conversely, if the missingness is informative, the base models are the mixed mode Bayesian mixture models presented in the previous section. Both approaches will associate each MTS $X^{(n)}$ with a q_2-dimensional posterior $\Pi^{(n)} \equiv \left(\pi_1^{(n)}, \ldots, \pi_{q_2}^{(n)}\right)^T$, where $\pi_g^{(n)}$ represents the probability that the MTS belongs to component g and q_2 is the total number of components in the base mixture model.

In ssTCK, label information is incorporated in an intermediate processing step in which the posteriors $\Pi^{(n)}$ are transformed, before the transformed posteriors are sent into Algorithms 2 and 3. More precisely, the transformation consists in mapping the posterior for the mixture components to a class "posterior" (probability), i.e. we seek to find a function $\mathcal{M} : [0,1]^{q_2} \to [0,1]^{N_c}$, $\Pi^{(n)} \xrightarrow{\mathcal{M}} \tilde{\Pi}^{(n)}$. Hence, we want to exploit the incomplete label information to find a transformation that merges the q_2 components of the mixture model into N_c clusters, where N_c is the number of classes.

The mapping $\mathcal{M}$ can be thought of as a (soft) N_c-class classifier, and hence there could be many possible ways of learning $\mathcal{M}$. However, choosing a too flexible classifier for this purpose leads to an increased risk of overfitting and could also unnecessarily increase

the algorithmic complexity. For these reasons, we restrict ourselves to searching for a linear transformation

$$\mathcal{M}(\Pi^{(n)}) = W^T \Pi^{(n)}, \quad W \in [0,1]^{q_2 \times N_c}. \tag{7}$$

Since the N_c-dimensional output $\tilde{\Pi}^{(n)} = \mathcal{M}(\Pi^{(n)})$ should represent a probability distribution, we add the constraint $\sum_{i=1}^{N_c} W_{ji} = 1,\ j = 1, \dots, q_2$.

A natural first step is to first assume that the label information is complete and look at the corresponding supervised kernel. In the following two subsections, we describe our proposed methods for learning the transformation $\mathcal{M}$ in supervised and semi-supervised settings, respectively.

4.1. Supervised time series cluster kernel (sTCK)

Supervised setting. Each base mixture model consists of q_2 components, and we assume that the number of components is greater or equal to the number of classes N_c. Further, assume that each MTS $X^{(n)}$ in the training set is associated with a N_c–dimensional one-hot vector $y^{(n)}$, which represents its label. Hence, the labels of the training set can be represented via a matrix $Y \in \{0,1\}^{N \times N_c}$, where N is the number of MTS in the training set.

We approach this problem by considering one component at the time. For a given component g, the task is to associate it with a class. One natural way to do this is to identify all members of component g and then simply count how many times each label occur. To account for class imbalance, one can then divide each count by the number of MTS in the corresponding class. One possible option would then be to assign the component to the class with the largest normalized count. However, by doing so, one is not accounting for uncertainty/disagreement within the component. Hence, a more elegant alternative is to simply use the normalized counts as the weights in the matrix W. Additionally, one has to account for that each MTS can simultaneously belong to several components, i.e. each MTS $X^{(n)}$ has a only soft membership to the component g, determined by the value $\pi_g^{(n)}$. This can be done using $\Pi^{(n)}$ as weights in the first step. This procedure is summarized in Algorithm 4.

Algorithm 4 Supervised posterior transformation.

Require: Posteriors $\{\Pi^{(n)}\}_{n=1}^N$ from mixture models consisting of q_2 components and labels $\{y^{(n)}\}_{n=1}^N$,
1: **for** $i = 1, \dots, q_2,\ j = 1, \dots, N_c$ **do**
2: Compute $W_{ij} = \frac{\sum_{n=1}^N y_j^{(n)} \pi_i^{(n)}}{\sum_{n=1}^N y_j^{(n)}}$.
3: $W_{ij} = \frac{W_{ij}}{\sum_{j=1}^{N_c} W_{ij}}$.
4: **end for**
5: Transform training and test posteriors via $\tilde{\Pi} = W^T \Pi$
Ensure: Transformed posteriors $\tilde{\Pi}^{(n)}$

4.2. Semi-supervised time series cluster kernel (ssTCK)

Setting Assume that the labels $\{y^{(n)}\}_{n=1}^L$, $L < N$, are known and $\{y^{(n)}\}_{n=L+1}^N$ are unknown.

In this setting, if one naively tries to apply Algorithm 4 based on only the labeled part of the dataset, one ends up dividing by 0s. The reason is that some of the components in the mixture model will contain only unlabeled MTS (the soft label analogy is that the probability that any of the labeled MTS belong to that particular component is zero or very close to zero). Hence, we need a way to assign labels to the components that do not contain any labeled MTS.

Note that each component is described by a probability distribution. A natural measure of dissimilarity between probability distributions is the Kullback–Leibler (KL) divergence [46]. Moreover, since the components are described by parametric distributions, the KL divergence has a simple closed-form expression. The KL divergence between two components, i and j, in our Bayesian GMM is given by

$$D_{KL}(f^{(i)} \parallel f^{(j)}) = \frac{1}{2}\Big(\sum_{v=1}^{V} \sum_{t=1}^{T} \sigma_{iv}^2 \sigma_{jv}^{-2} + \sigma_{jv}^{-2}(\mu_{jv}(t) - \mu_{iv}(t))^2 - 1 + \log(\sigma_{jv}^2) - \log(\sigma_{iv}^2) \Big), \tag{8}$$

where $f^{(i)} = f(X \mid R, \mu_i, \Sigma_i)$ is the density given in Eq. (4). The KL-divergence can be made symmetric via the transformation

$$D_{KL}^S(f^{(i)} \parallel f^{(j)}) = \frac{1}{2}\big(D_{KL}(f^{(i)} \parallel f^{(j)}) + D_{KL}(f^{(j)} \parallel f^{(i)})\big). \tag{9}$$

The underlying idea in our semi-supervised framework is to learn the transformation W for the clusters with only unlabeled points by finding the nearest cluster (in the D_{KL}^S-sense) that contain labeled points. This leads to Algorithm 5.

Algorithm 5 Semi-supervised posterior transformation.

Require: Posteriors $\{\Pi^{(n)}\}_{n=1}^N$ from mixture models consisting of q_2 components, labels $\{y^{(n)}\}_{n=1}^L$, and hyperparameter h.
1: **for** $i = 1, \dots, q_2,\ j = 1, \dots, N_c$ **do**
2: Compute $W_{ij} = \frac{\sum_{n=1}^N y_j^{(n)} \pi_i^{(n)}}{\sum_{n=1}^N y_j^{(n)}}$.
3: **end for**
4: **for all** k s.t. $\sum_{j=1}^{N_c} W_{kj} < h$ **do**
5: Let $\mathcal{L} = \{l \ s.t. \ \sum_{j=1}^{N_c} W_{lj} \geq h\}$
6: $W_{kj} = W_{lj}$ where $l = \arg\min_{l \in \mathcal{L}} D_{KL}^S(f^{(k)} \parallel f^{(l)})$.
7: **end for**
8: **for** $i = 1, \dots, q_2,\ j = 1, \dots, N_c$ **do**
9: $W_{ij} = \frac{W_{ij}}{\sum_{j=1}^{N_c} W_{ij}}$.
10: **end for**
11: Transform training or test posterior via $\tilde{\Pi} = W^T \Pi$
Ensure: Transformed posteriors $\tilde{\Pi}^{(n)}$

5. Experiments on synthetic and benchmark datasets

The experiments in this paper consists of two parts. The purpose of the first part was to demonstrate within a controlled environment situations where the proposed TCK_{IM} and ssTCK kernels might prove more useful than the TCK. In the second part (Section 6), we present a case study from a real-world medical application in which we compared to several baseline methods.

In the first part, we considered synthetic and benchmark datasets. The following experimental setup was considered. We performed kernel principal component analysis (KPCA) using time series cluster kernels and let the dimensionality of the embedding be 10. Thereafter, we trained a kNN-classifier with $k = 1$ on the embedding and evaluated performance in terms of classification accuracy on an independent test set. We let $Q = 30$ and

Table 1
Accuracy on the synthetic VAR(1) dataset.

	Unsupervised	Semi-supervised	Supervised
TCK	0.826	0.854	0.867
TCK_{IM}	0.933	0.967	0.970

Table 2
Description of benchmark time series datasets. Column 2 to 5 show the number of attributes, samples in training and test set, and number of classes, respectively. $T_{\min}$ is the length of the shortest MTS in the dataset and $T_{\max}$ the longest MTS. T is the length of the MTS after the transformation.

Datasets	Attributes	Train	Test	N_c	$T_{\min}$	$T_{\max}$	T	Source
uWave	3	200	4278	8	315	315	25	UCR
Char.Traj.	3	300	2558	20	109	205	23	UCI
Wafer	6	298	896	2	104	198	25	Olsz.
Japan.vow.	12	270	370	9	7	29	15	UCI

$\mathcal{I}_C = \{N_c, \dots, N_c + 20\}$. An additional hyperparameter h was introduced for ssTCK. We set h to 10^{-1} in our experiments. We also standardized each attribute to zero mean and unit standard deviation.

5.1. Synthetic example

To illustrate the effectiveness of the proposed methods, we first considered a controlled experiment in which a synthetic MTS dataset with two classes was sampled from a first-order vector autoregressive model,

$$\begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix} + \begin{pmatrix} \rho_1 & 0 \\ 0 & \rho_2 \end{pmatrix} \begin{pmatrix} x_1(t-1) \\ x_2(t-1) \end{pmatrix} + \begin{pmatrix} \xi_1(t) \\ \xi_2(t) \end{pmatrix} \tag{10}$$

To make $x_1(t)$ and $x_2(t)$ correlated with $\text{corr}(x_1(t), x_2(t)) = \rho$, we chose the noise term s.t., $\text{corr}(\xi_1(t), \xi_2(t)) = \rho\,(1 - \rho_1\rho_2)\,[(1 - \rho_1^2)(1 - \rho_2^2)]^{-1}$. For the first class ($y = 1$), we generated 100 two-variate MTS of length 50 for the training and 100 for the test, from the VAR(1)-model with parameters $\rho = \rho_1 = \rho_2 = 0.8$ and $\mathbb{E}[(x_1(t), x_2(t))^T \mid y = 1] = (0.5, -0.5)^T$. Analogously, the MTS of the second class ($y = 2$) were generated using parameters $\rho = -0.8$, $\rho_1 = \rho_2 = 0.6$ and $\mathbb{E}[(x_1(t), x_2(t))^T \mid y = 2] = (0, 0)^T$.

To simulate MNAR and inject informative missing patterns, we let $x_i^{(n)}(t)$ have a probability $p^{(n)}$ of being missing, given that $x_i^{(n)}(t) > -1$, $i = 1, 2$. We let $p^{(n)} = 0.9$ if $y^{(n)} = 1$ and $p_{(n)} = 0.8$ otherwise. By doing so, the missing ratio was roughly 63% in both classes.

Table 1 shows the accuracy on the test data for the different kernels. As expected, the TCK gives the lowest accuracy, 0.826. The ssTCK improves the accuracy considerably (0.854), and the supervised version (sTCK) gives further improvement (0.867). However, as we can see, the effect of explicitly modelling the missingness mechanism in the TCK_{IM} is larger. In this case the accuracy increases from 0.826 to 0.933. The two corresponding embeddings are plotted in Fig. 1(a) and (d), respectively. In the TCK embedding, there are many points from different classes that overlap with each other, whereas for the TCK_{IM} the number of overlapping points is much lower.

The ssTCK_{IM} improves the accuracy to 0.967 (from 0.933 for TCK_{IM} and 0.854 for ssTCK). The two embeddings obtained using the semi-supervised methods are shown in Fig. 1(b) and (e). The supervised version sTCK_{IM} yields a slight improvement in terms of accuracy compared to ssTCK_{IM} (0.970 vs. 0.967). Plots of the supervised embeddings are shown in Fig. 1(c) and (f). We can see that for the sTCK_{IM} the classes are clearly separated.

5.2. Performance of ssTCK on benchmark datasets

The purpose of the experiments reported in the following paragraph was to evaluate the impact of incorporating incomplete label information in the ssTCK. Towards that end, we considered benchmark datasets and artificially modified the number of labeled MTS in the training sets. We applied the proposed ssTCK to four MTS benchmark datasets from the UCR and UCI databases [47,48] and other published work [49], described in Table 2. Since some of the datasets contain MTS of varying length, we followed the approach of Wang et al. [50] and transformed all the MTS in the same dataset to the same length, T, determined by $T = \left\lceil \frac{T_{\max}}{\lceil \frac{T_{\max}}{25} \rceil} \right\rceil$, where $T_{\max}$ is the length of the longest MTS in the dataset and $\lceil\,\rceil$ is the ceiling operator. The number of labeled MTS was set to $\max\{20, 3 \cdot N_c\}$. ssTCK was compared to ordinary TCK and sTCK (assuming complete label information in the latter case).

Table 3
Classification accuracy for benchmark datasets obtained using TCK, ssTCK and sTCK.

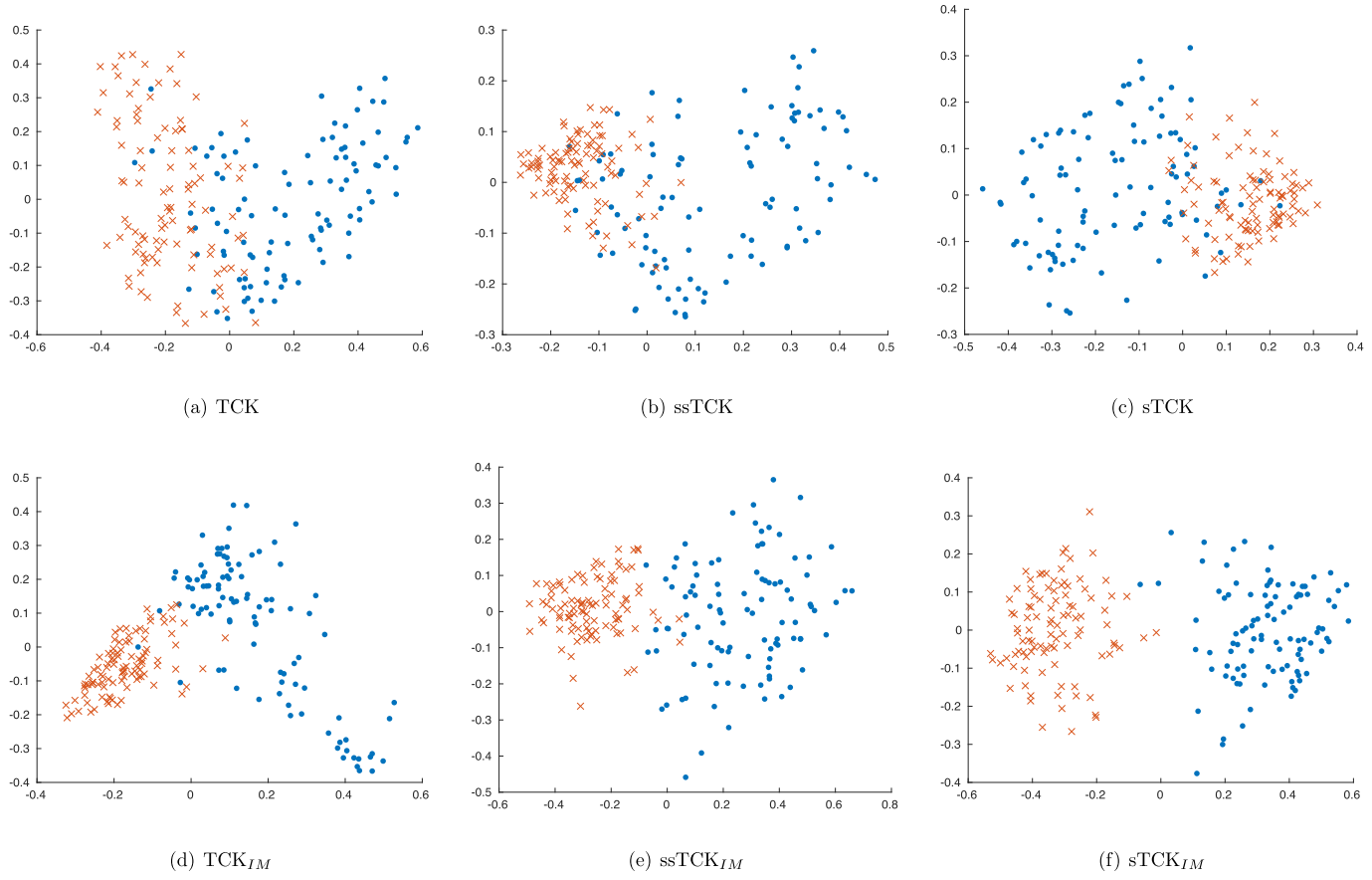
Datasets	TCK	ssTCK	sTCK
Char. Traj.	0.908	0.928	0.934
uWave	0.867	0.881	0.894
Wafer	0.956	0.970	0.970
Japanese vowels	0.946	0.962	0.968

Table 3 shows the performance of ssTCK for the 4 benchmark datasets. As we can see, compared to TCK, the accuracy in general increases using ssTCK. For the Wafer dataset, ssTCK yields the same performance as the supervised kernel. For the three other datasets, the performance of ssTCK is slightly worse than sTCK. These experiments demonstrate that ssTCK is capable of exploiting incomplete label information.

Further, we created 8 datasets by randomly removing 50% and 80%, respectively, of the values in each of the 4 benchmark datasets. As we can see from the results presented in Table 4, also in presence of missing data the accuracy in general increases using ssTCK, compared to TCK.

For comparison, in Table 4 we also added the results obtained using three other kernels; GAK, the linear kernel, and LPS. GAK and the linear kernel cannot process incomplete MTS and therefore we created complete datasets using mean imputation for these two kernels. LPS[2] was run using default hyperparameters, with the exception that we adjusted the segment length to be sampled from the interval $[6, 0.8T]$ to account for the relatively short MTS in our datasets. In accordance with [51], for GAK[3] we set the bandwidth σ to 0.1 times the median distance of all MTS in the training set scaled by the square root of the median length of all MTS, and the triangular parameter to 0.2 times the median length of all MTS. Distances were measured using the canonical metric induced by the Frobenius norm. In the linear kernel we set the constant c to 0. As we can see, the performance of these kernels is considerably worse than the time series cluster kernels for 7 out of 8 datasets. For uWave with 50% missingness, the performance of GAK and the linear kernel is similar to the TCK kernels.

[2] Matlab implementation: http://www.mustafabaydogan.com/.
[3] Matlab implementation: http://www.marcocuturi.net/GA.html.



(a) TCK (b) ssTCK (c) sTCK

(d) TCK_{IM} (e) ssTCK_{IM} (f) sTCK_{IM}

Fig. 1. Plot of the two-dimensional KPCA representation of the synthetic data obtained using 6 different time series cluster kernels. The datapoints are colour-coded according to their labels.

Table 4
Classification accuracy for benchmark datasets obtained using TCK, ssTCK and sTCK.

Missing rate	Datasets	TCK	ssTCK	sTCK	GAK	Linear	LPS
50%	Char. Traj.	0.751	0.780	0.797	0.588	0.589	0.127
	uWave	0.812	0.834	0.850	0.828	0.813	0.411
	Wafer	0.956	0.970	0.972	0.792	0.791	0.823
	Japanese vowels	0.929	0.948	0.947	0.827	0.824	0.746
80%	Char. Traj.	0.282	0.310	0.331	0.194	0.192	0.062
	uWave	0.589	0.592	0.603	0.441	0.464	0.234
	Wafer	0.926	0.934	0.934	0.796	0.805	0.819
	Japanese vowels	0.809	0.836	0.847	0.473	0.489	0.389

5.3. Exploiting informative missingness in benchmark datasets

To evaluate the effect of modelling the missing patterns in TCK_{IM}, we generated 8 versions of the Wafer and Japanese vowels datasets by manually injecting missing elements using the following procedure. For each attribute $v \in \{1, \dots, V\}$, a number $c_v \in \{-1, 1\}$ was randomly sampled with equal probabilities. If $c_v = 1$, the attribute v is positively correlated with the labels, otherwise negatively correlated. For each MTS $X^{(n)}$ and attribute, a missing rate γ_{nv} was sampled from the uniform distribution $\mathcal{U}[0.3 + E \cdot c_v \cdot (y^{(n)} - 1), 0.7 + E \cdot c_v \cdot (y^{(n)} - 1)]$. This ensures that the overall missing rate of each dataset is approximately 50%. $y^{(n)} \in \{1, \dots N_c\}$ is the label of the MTS $X^{(n)}$ and E is a parameter, which we tune for each dataset in such a way that the absolute value of the Pearson correlation between the missing rates for the attributes γ_v and the labels $y^{(n)}$ takes the values $\{0.2, 0.4, 0.6, 0.8\}$, respectively. The higher the value of the Pearson correlation, the higher is the informative missingness.

Table 5 shows the performance of the proposed TCK_{IM} and three baseline models (TCK, TCK_B, and TCK_0). The first baseline is ordinary TCK, which ignores the missingness mechanism. For the Wafer dataset, the performance of this baseline was quite similar across all four settings. For the Japanese vowels dataset, the performance actually decreases as the information in the missing patterns increases. In the second baseline, TCK_B, we tried to model the missing patterns by concatenating the binary missing indicator MTS R to the MTS X and creating a new MTS with $2V$ attributes. Then, we trained ordinary TCK on this representation. For the Wafer dataset, the performance decreases considerably as the informative missingness increases. For the Japanese vowels, this baseline yields the best performance when the correlation is 20%. However, the performance actually decreases as the informative missingness increases. Hence, informative missingness is not captured with this baseline. In the last baseline, TCK_0, we investigated if it is possible to capture informative missingness by imputing zeros for the missing values and then training the TCK on the im-

Table 5
Classification accuracy on benchmark datasets that contain missing data.

Correlation	TCK	TCK_B	TCK_0	TCK_{IM}	TCK	TCK_B	TCK_0	TCK_{IM}
	Wafer				**Japanese vowels**			
0.2	0.951	0.951	0.951	**0.955**	0.938	**0.954**	0.951	0.940
0.4	**0.961**	0.953	0.955	**0.961**	0.932	0.938	0.938	**0.941**
0.6	0.961	0.900	0.965	**0.996**	0.922	0.946	0.924	**0.962**
0.8	0.958	0.893	0.963	**1.000**	0.922	0.924	0.935	**0.968**
	uWave				**Character trajectories**			
0.2	0.763	0.457	0.755	**0.841**	**0.854**	0.742	0.847	0.851
0.4	0.807	0.587	0.813	**0.857**	0.851	0.788	0.842	**0.867**
0.6	0.831	0.674	0.837	**0.865**	0.825	0.790	0.824	**0.871**
0.8	0.834	0.699	0.844	**0.884**	0.839	0.707	0.853	**0.901**

puted data. This baseline yields similar performance across all 4 settings for the Wafer dataset, and for Japanese vowels, TCK_0 has a similar behaviour as TCK_B, i.e. it does not capture informative missing patterns. The proposed TCK_{IM} achieves the best accuracy for 7 out of 8 settings and has the expected behaviour, namely that the accuracy increases as the correlation between missing values and class labels increases. The performance is similar to TCK when the amount of information in the missing patterns is low, whereas TCK is clearly outperformed when the informative missingness is high. This demonstrates that TCK_{IM} effectively utilizes informative missing patterns.

To also test if TCK_{IM} is capable of exploiting other types of informative missingness, we generated 8 versions of uWave and Character trajectories datasets using the following approach. Both of these datasets consists of 3 attributes. For each attribute $v \in \{1, \dots, V\}$, a number $c_v \in \{-1, 1\}$ was randomly sampled with equal probabilities. For the attribute(s) with $c_v = -1$, we let it be negatively correlated with the labels by sampling the missing rate γ_{nv} from $\mathcal{U}[0.7 - E \cdot (y^{(n)} - 1), 1 - E \cdot (y^{(n)} - 1)]$. For the attribute with $c_v = 1$, we let it be positively correlated with the labels by sampling the missing rate γ_{nv} from $\mathcal{U}[0.3 + E \cdot (y^{(n)} - 1), 0.6 + E \cdot (y^{(n)} - 1)]$. We let each element with $x_v^{(n)}(t) > \mu_v$ have a probability γ_{nv} of being missing, where μ_v is the mean of attribute v computed over the complete dataset. The fact that the probability of being missing depends on the missing values means that, within each class, the missingness mechanism is MNAR. We tuned the parameter E such that the mean absolute value of the Pearson correlation between γ_v and the labels took the values $\{0.2, 0.4, 0.6, 0.8\}$. By doing so, the overall missing rate was approximately 32% for uWave and 45% for the Characters. However, we note that in this case the overall missing rate varies slightly as a function of the Pearson correlation.

Table 5 shows the performance on the 8 datasets created from uWave and Char. traj. One thing to notice here is the poor performance of TCK_B. This demonstrates the importance of using the mixed mode mixtures to model the two modalities in $U = (X, R)$. To naively apply TCK based on the GMMs to the concatenated MTS do not provide the desired performance. Further, we see that TCK_{IM} achieves the best accuracy for 7 out of 8 settings and the accuracy increases as the correlation increases. For the Characters, the performance of TCK_{IM} is similar to TCK for low correlation but increases as the missingness information increases, whereas the performance of TCK actually decreases. One possible explanation is that for this dataset, two of the variables were positively correlated with the labels and therefore the missing rate increases with increasing correlation. Regarding the results for uWave, it is a bit surprising that the largest difference in performance between TCK and TCK_{IM} occurs when the correlation is lowest. There might be several reasons to this: a peculiarity of the dataset and/or that the MNAR missingness created missing patterns that negatively affect TCK.

6. Case study: detecting infections among patients undergoing colon rectal cancer surgery

In this case study, the focus was to detect Surgical Site Infection (SSI), which is one of the most common types of nosocomial infections [52] and represents up to 30% of hospital-acquired infections [53,54]. The importance of the topic of SSI prediction is reflected in several recent initiatives. For instance, the current study is part of a larger research effort by the current team, on SSI prediction and detection of postoperative adverse events related to gastrointestinal surgery within the context of improving the *quality of surgery* [25,28,55–58]. Clearly, the reason for this massive interest is that a reduction in the number of postoperative complications such as SSI will be of great benefit both for the patients and for the society.

Many studies have shown that laboratory tests, and blood tests in particular, are especially important predictors for SSI, both pre- and post-operatively [56,57,59–67]. Therefore, blood tests provided the basis also for this case study.

6.1. Data collection

Ethics approval for the parent study was obtained from the Data Inspectorate and the Ethics Committee at the University Hospital of North Norway (UNN) [58]. In [58], a cohort consisting of 7741 patients was identified by extracting the electronic health records for all patients that underwent a gastrointestinal surgical procedure at UNN in the years 2004–2012. In this case study, we were particularly interested in detecting SSI, which is an infection particularly associated with colorectal cancer surgery [68]. Therefore, patients who did not undergo this type of surgery were excluded, reducing the size of the cohort to 1137 patients.

In collaboration with a clinician (author A. R.), we extracted data for 11 of the most common blood tests from the patient's EHRs. The value of a patient's blood test, e.g. his or hers hemoglobin level, can be considered as a continuous variable over time. However, blood tests are usually measured on a daily basis, and therefore, for the purpose of the current analysis, we discretized time and let each time interval be one day. Hence, the blood samples could naturally be represented as MTS and needed no further feature preprocessing in our framework.

All blood tests were not available every day for each patient, which means that the dataset contained missing data, and we expected the missing patterns to be informative since whether a test is performed depends on whether the doctor thinks it is needed. We focused on detection of SSI within 10 days after surgery and therefore the length of the time series is 10. Patients with no recorded lab tests during the period from postoperative day 1 until day 10 were removed from the cohort, which lead to a final cohort consisting of 858 patients. The average proportion of missing data

Table 6
List of extracted blood tests and their corresponding missing rates.

Attribute nr.	Blood test	Missing rate
1	Hemoglobin	0.646
2	Leukocytes	0.727
3	C-Reactive Protein	0.691
4	Potassium	0.709
5	Sodium	0.712
6	Creatinine	0.867
7	Thrombocytes	0.921
8	Albumin	0.790
9	Carbamide	0.940
10	Glucose	0.921
11	Amylase	0.952

in the cohort was 80.7%. Table 6 shows a list of the blood tests we considered in this study and their corresponding missing rate.

Guided by input from clinicians, the International Classification of Diseases (ICD10) or NOMESCO Classification of Surgical Procedures (NCSP) codes related to severe postoperative complications were considered to identify the patients in the cohort that developed postoperative SSI. Patients that did not have these codes and did not have the word "infection" in any of their postoperative text documents were considered as controls. This lead to a dataset with 227 infected patients (cases) and 631 non-infected patients (control).

6.2. Experimental setup

The objective of this case study was to evaluate how the proposed MTS kernels perform in a real-world application from medicine. We would like to emphasize that the proposed kernels are mainly designed for situations when there are no, or only a few, ground-truth labels available. However, in order to evaluate the quality of these kernels, we adopted a supervised scheme. Hence, we followed the scheme presented in Fig. 2, i.e. we computed the kernel from the MTS representations of the blood tests and performed KPCA, followed by kNN classification in the KPCA space. We set the dimensionality of the KPCA-representation to 10 in all experiments. The number of neighbors k was set using 5-fold cross validation.

Four baseline kernels were considered, namely TCK, LPS, GAK and the linear kernel. GAK and the linear kernel cannot work on incomplete datasets, and therefore, we created 2 complete datasets using mean and LOCF imputation. In order to investigate if it is possible to better exploit the information from the missing patterns for the LPS, GAK and linear kernels, we also created baselines by concatenating the binary indicator MTS $R^{(n)}$ to the MTS $X^{(n)}$.

We performed 5-fold cross validation and reported results in terms of F1-score, sensitivity, specificity and accuracy. Sensitivity is the fraction of actual positives (has SSI) correctly classified as positive, whereas specificity is the fraction of actual negatives that are correctly classified as negative. F1-score is the harmonic mean of precision and sensitivity, where precision is the fraction of actual positives among all those that are classified as positive cases.

6.3. Results

Table 7 shows the performance in terms of 4 evaluation metrics for 11 baseline kernels as well as the proposed TCK_{IM} kernel on the task of detecting patients suffering from SSI. We see that the kernels that rely on imputation performs much worse than other kernels in terms of F1-score, sensitivity and accuracy. These methods do, however, achieve a high specificity. However, any classifier can achieve a specificity of 1 simply by classifying all cases as negative, but this of course leads to lower F1-score and sensitivity. The main reasons why these methods do not perform better are probably that the imputation methods introduce strong biases into the data and that the missingness mechanism is ignored. The TCK and LPS kernels perform quite similarly across all 4 evaluation metrics (LPS slightly better). The F1-score, sensitivity and accuracy achieved for these methods are considerably higher than the corresponding scores for the GAK and linear kernel. One of the reasons why these methods perform better than the imputation methods is that ignoring the missingness leads to lower bias than replacing missing values with biased estimates. The performance of the linear kernel and GAK improves a bit by accounting for informative missingness, whereas the performance of LPS decreases. TCK_{IM} performs similarly to the baselines in terms of specificity, but considerably better in terms of F1-score, sensitivity and accuracy. This demonstrates that the missing patterns in the blood test time series are informative and the TCK_{IM} is capable of exploiting this information to improve performance on the task of detecting patients with infections.

Fig. 3 shows KPCA embeddings corresponding to the two largest eigenvalues obtained using 5 different kernels. While the representations obtained using GAK and the linear kernel are noisy and to a large degree mix the infected and non-infected patients, the two

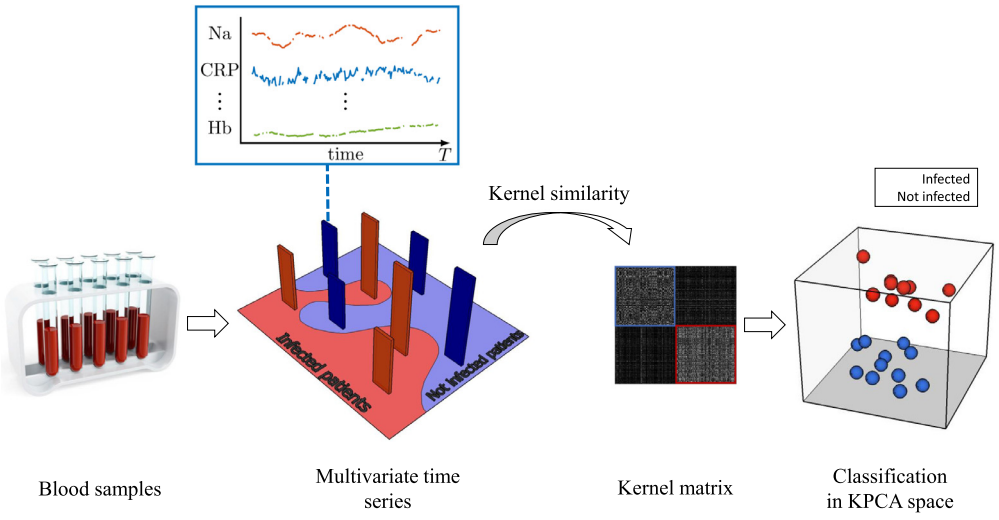
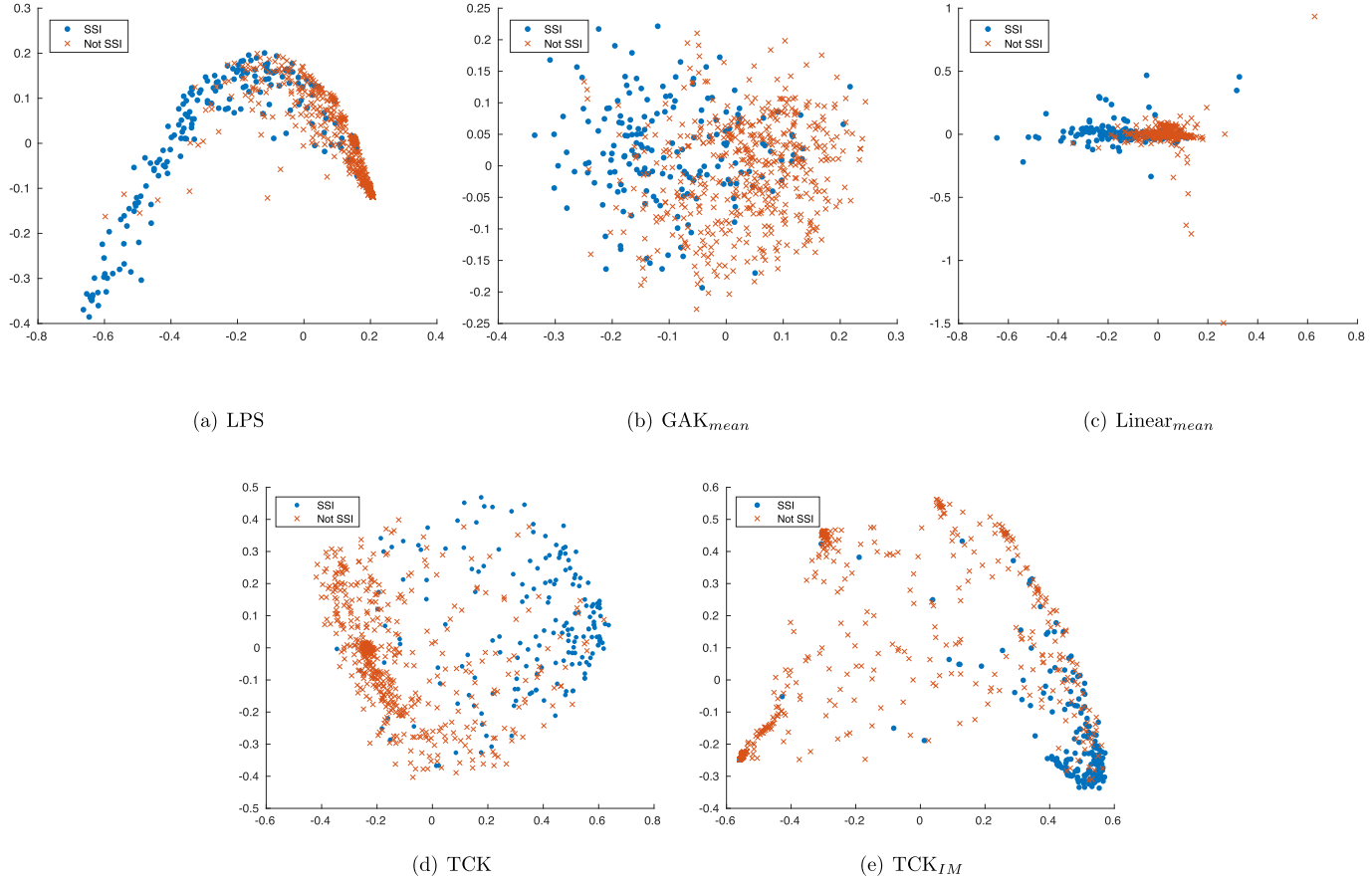


Fig. 2. Overview of the approach taken to detect postoperative SSI from MTS blood samples.

Table 7
Performance (mean ± se) on the SSI dataset. The best results are in bold.

	Kernel	F1-score	Sensitivity	Specificity	Accuracy
Ignore missingness	TCK	0.726 ± 0.045	0.678 ± 0.035	0.930 ± 0.024	0.863 ± 0.023
	LPS	0.746 ± 0.035	0.696 ± 0.056	0.939 ± 0.019	0.875 ± 0.016
Impute	GAK_{LOCF}	0.570 ± 0.045	0.484 ± 0.059	0.924 ± 0.022	0.808 ± 0.017
	GAK_{mean}	0.629 ± 0.046	0.502 ± 0.059	0.966 ± 0.023	0.843 ± 0.016
	$Linear_{LOCF}$	0.557 ± 0.058	0.480 ± 0.073	0.914 ± 0.017	0.800 ± 0.018
	$Linear_{mean}$	0.599 ± 0.030	0.489 ± 0.041	0.948 ± 0.043	0.826 ± 0.024
Informative	LPS_{IM}	0.720 ± 0.062	0.661 ± 0.069	0.937 ± 0.036	0.863 ± 0.032
	$GAK_{IM+LOCF}$	0.669 ± 0.015	0.586 ± 0.024	0.940 ± 0.021	0.846 ± 0.011
	$GAK_{IM+mean}$	0.696 ± 0.030	0.617 ± 0.033	0.945 ± 0.022	0.856 ± 0.011
	$Linear_{IM+LOCF}$	0.628 ± 0.016	0.529 ± 0.030	0.945 ± 0.011	0.834 ± 0.005
	$Linear_{IM+mean}$	0.668 ± 0.037	0.568 ± 0.033	$\mathbf{0.951 \pm 0.030}$	0.850 ± 0.021
	TCK_{IM}	$\mathbf{0.802 \pm 0.016}$	$\mathbf{0.806 \pm 0.027}$	0.927 ± 0.017	$\mathbf{0.895 \pm 0.010}$



(a) LPS (b) GAK_{mean} (c) $Linear_{mean}$

(d) TCK (e) TCK_{IM}

Fig. 3. Plot of the two-dimensional KPCA representation of the colon rectal cancer surgery patients obtained using 5 kernels.

classes (SSI and non-SSI) are more separated in the representations obtained using TCK and LPS. The TCK_{IM} is even better at forcing the SSI patients to stay in the same region or cluster while it at the same time spreads out the patients without infection, revealing the diversity among these patients.

7. Conclusions and future directions

In this work, we presented robust multivariate time series kernels capable of exploiting informative missing patterns and incomplete label information. In contrast to other frameworks that exploit informative missingness [5,21], which need complete label information, the time series cluster kernels are specially designed for situations in which no labels or only a few labels are available. Lack of labels and large amounts of missing data are two challenges that characterize the medical domain, and therefore, we think the proposed kernels will be particularly useful in this domain, which we also demonstrated in this work through a case study of postoperative infections among colon rectal cancer patients. However, the kernels are not limited to this domain. We believe that these kernels could be useful tools in other application domains facing similar challenges.

A limitation of TCK_{IM} is that if the missingness is by no means correlated with the outcome of interest, there will be limited gain in performance compared to the TCK, or might even a decrease in performance. For this reason it is important that the user has some domain knowledge and has some understanding about the process that led to missing values in the data, as illustrated in our case study from healthcare.

An other limitation of the time series cluster kernels is that they are designed for MTS of the same length. A possible next step would be to work on a formulation that can deal with vary-

ing length. In further work, we would also like to investigate the possibility of introducing a Bayesian formulation for the discrete modality in the mixed mode mixture models by putting informative priors over the parameters in the Bernoulli part of the model.

Declaration of Competing Interest

Authors declare that they have no conflict of interest.

Acknowledgements

This work was partially funded by the Norwegian Research Council FRIPRO grant no. 239844 on developing the *Next Generation Learning Machines*. Cristina Soguero-Ruiz is partially supported by project PID2019-107768RA-I00 (AAVis-BMR) from Spanish Government, by project DTS17/00158 from Institute of Health Carlos III (Spain), by project Ref. F656 from Rey Juan Carlos University and by the Young R&D Project Ref. 2020-661 financed by Rey Juan Carlos University and Community of Madrid.

The authors would like to thank Kristian Hindberg from UiT The Arctic University of Norway for his assistance on preprocessing and extracting the data from the EHR system. We would also like to thank Rolv-Ole Lindsetmo and Knut Magne Augestad from the University Hospital of North Norway, Fred Godtliebsen from UiT, together with Stein Olav Skrøvseth from the Norwegian Centre for E-health Research for helpful discussions throughout the study and manuscript preparation.

References

[1] D.B. Rubin, Inference and missing data, Biometrika 63 (3) (1976) 581–592.
[2] G. Molenberghs, Incomplete data in clinical studies: analysis, sensitivity, and sensitivity analysis, Drug Inf. J. 43 (4) (2009) 409–429.
[3] G. Molenberghs, C. Beunckens, C. Sotto, M.G. Kenward, Every missingness not at random model has a missingness at random counterpart with equal fit, J. R. Stat. Soc. 70 (2) (2008) 371–388.
[4] A.S. Allen, P.J. Rathouz, G.A. Satten, Informative missingness in genetic association studies: case-parent designs, Am. J. Hum. Genet. 72 (3) (2003) 671–680.
[5] Z. Che, S. Purushotham, K. Cho, D. Sontag, Y. Liu, Recurrent neural networks for multivariate time series with missing values, Sci. Rep. 8 (1) (2018) 6085.
[6] N. Fouladgar, K. Främling, A novel LSTM for multivariate time series with massive missingness, Sensors 20 (10) (2020) 2832.
[7] B. Yang, M. Ye, Q. Tan, P.C. Yuen, Cross-domain missingness-aware time-series adaptation with similarity distillation in medical applications, IEEE Trans. Cybern. (2020) 1–14, doi:10.1109/TCYB.2020.3011934.
[8] J.L. Schafer, J.W. Graham, Missing data: our view of the state of the art, Psychol. Methods 7 (2) (2002) 147.
[9] R.J.A. Little, D.B. Rubin, Statistical Analysis with Missing Data, John Wiley & Sons, 2014.
[10] W. Weihan, Magan: a masked autoencoder generative adversarial network for processing missing IoT sequence data, Pattern Recognit. Lett. 138 (2020) 211–216.
[11] S.C.-X. Li, B. Marlin, Learning from irregularly-sampled time series: amissing data perspective, in: Proceedings of the 37th International Conference on Machine Learning, ACM, 2020, pp. 5756–5765.
[12] P.J. García-Laencina, J.-L. Sancho-Gómez, A.R. Figueiras-Vidal, Pattern classification with missing data: a review, Neural Comput. Appl. 19 (2) (2010) 263–282.
[13] S.A. Rahman, Y. Huang, J. Claassen, N. Heintzman, S. Kleinberg, Combining Fourier and lagged k-nearest neighbor imputation for biomedical time series data, J. Biomed. Inform. 58 (2015) 198–207.
[14] J.M. Engels, P. Diehr, Imputation of missing longitudinal data: a comparison of methods, J. Clin. Epidemiol. 56 (10) (2003) 968–976.
[15] I.R. White, P. Royston, A.M. Wood, Multiple imputation using chained equations: issues and guidance for practice, Stat. Med. 30 (4) (2011) 377–399.
[16] F.M. Bianchi, L. Livi, K.O. Mikalsen, M. Kampffmeyer, R. Jenssen, Learning representations of multivariate time series with missing data, Pattern Recognit. 96 (2019) 106973.
[17] X. Tang, H. Yao, Y. Sun, C.C. Aggarwal, P. Mitra, S. Wang, Joint modeling of local and global temporal dynamics for multivariate time series forecasting with missing values, in: Proceedings of the AAAI Conference on Artificial Intelligence, 34, 2020, pp. 5956–5963.
[18] Q. Tan, M. Ye, B. Yang, S. Liu, A.J. Ma, T.C.-F. Yip, G.L.-H. Wong, P. Yuen, DATA–GRU: dual-attention time-aware gated recurrent unit for irregular multivariate time series, in: Proceedings of the AAAI Conference on Artificial Intelligence, 34, 2020, pp. 930–937.
[19] Q. Li, Y. Xu, VS-GRU: a variable sensitive gated recurrent neural network for multivariate time series with massive missing values, Appl. Sci. 9 (15) (2019) 3041.
[20] K.O. Mikalsen, F.M. Bianchi, C. Soguero-Ruiz, S.O. Skrøvseth, R.-O. Lindsetmo, A. Revhaug, R. Jenssen, Learning similarities between irregularly sampled short multivariate time series from EHRs, 3rd ICPR International Workshop on Pattern Recognition for Healthcare Analytics, Cancun, Mexico, 2016.
[21] Z.C. Lipton, D. Kale, R. Wetzel, Directly modeling missing data in sequences with RNNs: improved classification of clinical time series, in: Machine Learning for Healthcare Conference, 56, PMLR, 2016, pp. 253–270.
[22] M. Ghassemi, M.A.F. Pimentel, T. Naumann, T. Brennan, D.A. Clifton, P. Szolovits, M. Feng, A multivariate timeseries modeling approach to severity of illness assessment and forecasting in ICU with sparse, heterogeneous clinical data, in: Conference on Artificial Intelligence, AAAI, 2015, pp. 446–453.
[23] C. Sun, S. Hong, M. Song, H. Li, A review of deep learning methods for irregularly sampled medical time series data, arXiv preprint arXiv:2010.12493(2020).
[24] K.O. Mikalsen, F.M. Bianchi, C. Soguero-Ruiz, R. Jenssen, Time series cluster kernel for learning similarities between multivariate time series with missing data, Pattern Recognit. 76 (2018) 569–581.
[25] K.O. Mikalsen, C. Soguero-Ruiz, A. Revhaug, R.-O. Lindsetmo, R. Jenssen, et al., Using anchors from free text in electronic health records to diagnose postoperative delirium, Comput. Methods Prog. Biomed. 152 (Supplement C) (2017) 105–114.
[26] R. Jenssen, Kernel entropy component analysis, IEEE Trans. Pattern Anal. Mach. Intell. 33 (5) (2010) 847–860.
[27] G. Camps-Valls, L. Bruzzone, Kernel Methods for Remote Sensing Data Analysis, John Wiley & Sons, 2009.
[28] C. Soguero-Ruiz, A. Revhaug, R.-O. Lindsetmo, K.M. Augestad, R. Jenssen, et al., Support vector feature selection for early detection of anastomosis leakage from bag-of-words in electronic health records, IEEE J. Biomed. Health Inform. 20 (5) (2016) 1404–1415.
[29] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, 2004.
[30] F.J. Király, H. Oberhauser, Kernels for sequentially ordered data, J. Mach. Learn. Res. 20 (31) (2019) 1–45.
[31] H. Chen, F. Tang, P. Tino, X. Yao, Model-based kernel for efficient time series analysis, in: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2013, pp. 392–400.
[32] D.J. Berndt, J. Clifford, Using dynamic time warping to find patterns in time series, in: Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining, AAAI Press, 1994, pp. 359–370.
[33] H. Deng, W. Chen, Q. Shen, A.J. Ma, P.C. Yuen, G. Feng, Invariant subspace learning for time series data based on dynamic time warping distance, Pattern Recognit. 102 (2020) 107210.
[34] B.K. Iwana, S. Uchida, Time series classification using local distance-based features in multi-modal fusion networks, Pattern Recognit. 97 (2020) 107024.
[35] P.-F. Marteau, S. Gibet, On recursive edit distance kernels with application to time series classification, IEEE Trans. Neural Netw. Learn. Syst. 26 (6) (2015) 1121–1133.
[36] M. Cuturi, J.-P. Vert, O. Birkenes, T. Matsui, A kernel for time series based on global alignments, in: Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on, 2, IEEE, 2007, pp. II–413.
[37] M. Cuturi, Fast global alignment kernels, in: Proceedings of the 28th International Conference on Machine Learning, 2011, pp. 929–936.
[38] M.G. Baydogan, G. Runger, Time series representation and similarity based on local autopatterns, Data Min. Knowl. Discov. 30 (2) (2016) 476–509.
[39] A. Barla, F. Odone, A. Verri, Histogram intersection kernel for image classification, in: Proceedings of International Conference on Image Processing, 3, IEEE, 2003, pp. III–513.
[40] S. Akodad, L. Bombrun, Y. Berthoumieu, C. Germain, Cluster kernel for learning similarities between symmetric positive definite matrix time series, in: 2020 IEEE International Conference on Image Processing (ICIP), IEEE, 2020, pp. 3304–3308.
[41] T.G. Dietterich, Ensemble methods in machine learning, in: International Workshop on Multiple Classifier Systems, Springer Berlin Heidelberg, 2000, pp. 1–15.
[42] L.K. Hansen, P. Salamon, Neural network ensembles, IEEE Trans. Pattern Anal. Mach. Intell. 12 (10) (1990) 993–1001.
[43] S. Vega-Pons, J. Ruiz-Shulcloper, A survey of clustering ensemble algorithms, Int. J. Pattern Recognit. Artif. Intell. 25 (03) (2011) 337–372.
[44] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. Ser. B (1977) 1–38.
[45] G. McLachlan, T. Krishnan, The EM Algorithm and Extensions, 382, John Wiley & Sons, 2007.
[46] S. Kullback, R.A. Leibler, On information and sufficiency, Ann. Math. Stat. 22 (1) (1951) 79–86.
[47] H.A. Dau, E. Keogh, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C.A. Ratanamahatana, Yanping, B. Hu, N. Begum, A. Bagnall, A. Mueen, G. Batista, The UCR time series classification archive, 2018, https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.
[48] M. Lichman, UCI machine learning repository, 2013, (http://archive.ics.uci.edu/ml). Accessed: 2018-08-29.
[49] R.T. Olszewski, Generalized Feature Extraction for Structural Pattern Recognition in Time-series Data, Carnegie Mellon University, Pittsburgh, PA, USA, 2001 Ph.D. thesis.
[50] L. Wang, Z. Wang, S. Liu, An effective multivariate time series classification approach using echo state network and adaptive differential evolution algorithm, Expert Syst. Appl. 43 (2016) 237–249.

[51] Fast global alignment kernel Matlab implementation, (http://www.marcocuturi.net/GA.html). Accessed: 2018-08-02.
[52] S.S. Lewis, R.W. Moehring, L.F. Chen, D.J. Sexton, D.J. Anderson, Assessing the relative burden of hospital-acquired infections in a network of community hospitals, Infect. Control Hosp. Epidemiol. 34 (11) (2013) 1229–1230.
[53] S.S. Magill, W. Hellinger, J. Cohen, R. Kay, et al., Prevalence of healthcare-associated infections in acute care hospitals in Jacksonville, Florida, Infect. Control 33 (03) (2012) 283–291.
[54] G. de Lissovoy, K. Fraeman, V. Hutchins, D. Murphy, D. Song, B.B. Vaughn, Surgical site infection: incidence and impact on hospital utilization and treatment costs, Am. J. Infect. Control 37 (5) (2009) 387–397.
[55] C. Soguero-Ruiz, A. Revhaug, R.-O. Lindsetmo, R. Jenssen, et al., Predicting colorectal surgical complications using heterogeneous clinical data and kernel methods, J. Biomed. Inform. 61 (2016) 87–96.
[56] A.S. Strauman, F.M. Bianchi, K.O. Mikalsen, M. Kampffmeyer, C. Soguero-Ruiz, R. Jenssen, Classification of postoperative surgical site infections from blood measurements with missing data using recurrent neural networks, in: 2018 IEEE EMBS International Conference on Biomedical Health Informatics (BHI), 2018, pp. 307–310.
[57] C. Soguero-Ruiz, R. Jenssen, K.M. Augestad, S.O. Skrøvseth, et al., Data–driven temporal prediction of surgical site infection, in: AMIA Annual Symposium Proceedings, 2015, American Medical Informatics Association, 2015, p. 1164.
[58] K. Jensen, C. Soguero-Ruiz, K.O. Mikalsen, R.-O. Lindsetmo, I. Kouskoumvekaki, M. Girolami, S.O. Skrovseth, K.M. Augestad, Analysis of free text in electronic health records for identification of cancer patient trajectories, Sci. Rep. 7 (2017) 46226.
[59] J. Silvestre, J. Rebanda, C. Lourenço, P. Póvoa, Diagnostic accuracy of C-reactive protein and procalcitonin in the early detection of infection after elective colorectal surgery–a pilot study, BMC Infect. Dis. 14 (1) (2014) 444.
[60] F.J. Medina-Fernández, D.J. Garcilazo-Arismendi, R. García-Martín, L. Rodríguez-Ortiz, J. Gómez-Barbadillo, et al., Validation in colorectal procedures of a useful novel approach for the use of C-reactive protein in postoperative infectious complications, Colorectal Dis. 18 (3) (2016) O111–O118.
[61] M.R. Angiolini, F. Gavazzi, C. Ridolfi, M. Moro, P. Morelli, M. Montorsi, A. Zerbi, Role of C-reactive protein assessment as early predictor of surgical site infections development after pancreaticoduodenectomy, Dig. Surg. 33 (4) (2016) 267–275.
[62] S. Liu, J. Miao, G. Wang, M. Wang, X. Wu, K. Guo, M. Feng, W. Guan, J. Ren, Risk factors for postoperative surgical site infections in patients with Crohn's disease receiving definitive bowel resection, Sci. Rep. 7 (1) (2017) 9828.
[63] E. Mujagic, W.R. Marti, M. Coslovsky, J. Zeindler, et al., The role of preoperative blood parameters to predict the risk of surgical site infection, Am. J. Surg. 215 (4) (2018) 651–657.
[64] A. Goulart, C. Ferreira, A. Estrada, F. Nogueira, S. Martins, A. Mesquita-Rodrigues, N. Sousa, P. Leao, Early inflammatory biomarkers as predictive factors for freedom from infection after colorectal cancer surgery: a prospective cohort study, Surg. Infect. 19 (4) (2018) 446–450.
[65] Z. Hu, G.B. Melton, E.G. Arsoniadis, Y. Wang, M.R. Kwaan, G.J. Simon, Strategies for handling missing clinical data for automated surgical site infection detection from the electronic health record, J. Biomed. Inform. 68 (2017) 112–120.
[66] S.L. Gans, J.J. Atema, S. Van Dieren, B.G. Koerkamp, M.A. Boermeester, Diagnostic value of C-reactive protein to rule out infectious complications after major abdominal surgery: a systematic review and meta-analysis, Int. J. Colorectal Dis. 30 (7) (2015) 861–873.
[67] P.C. Sanger, G.H. van Ramshorst, E. Mercan, et al., A prognostic model of surgical site infection using daily clinical wound assessment, J. Am. Coll. Surg. 223 (2) (2016) 259–270.e2.
[68] E.H. Lawson, C.Y. Ko, J.L. Adams, W.B. Chow, B.L. Hall, Reliability of evaluating hospital quality by colorectal surgical site infection type, Ann. Surg. 258 (6) (2013) 994–1000.

Karl Øyvind Mikalsen received the M.Sc degree in Applied Mathematics in 2014 at UiT The Arctic University of Norway, Tromsø, Norway, where he currently is working towards a Ph.D. degree in Machine Learning. His research interests include machine learning for healthcare, time series analysis, kernel methods, unsupervised and weakly supervised learning.

Cristina Soguero-Ruiz got the Ph.D. in 2015, awarded with the Orange Best Ph.D. Award, at Rey Juan Carlos University, where she is an assistant professor. In addition, she is an associate member in the Machine Learning Group at University of Tromsφ. Her research interests include machine learning and healthcare analytics.

Filippo Bianchi received the B.Sc. in Computer Engineering (2009), the M.Sc. in Artificial Intelligence and Robotics (2012) and Ph.D. in Machine Learning (2016) from Sapienza University. He worked 2 years as research assistant at Ryerson University. He's currently a postdoc at UiT. Research interests include graph-matching, reservoir computing and deep-learning.

Arthur Revhaug MD Ph.D. is Professor of Surgery at UiT The Arctic University of Norway and the University Hospital of North-Norway. He is a funding member of the ERAS Society and has a long standing interest in how to use machine learning methods useful in clinical practice.

Robert Jenssen directs the UiT Machine Learning Group: https://machine-learning.uit.no/. The group is advancing research on deep learning and kernel machines, as well as healthcare analytics, remote sensing, and industrial applications. Jenssen is an associate editor of Pattern Recognition, an IEEE TC MLSP member, and on the IAPR Governing Board.