51st IEEE Conference on Decision and Control
December 10-13, 2012. Maui, Hawaii, USA

Learning from Time Series: Supervised Aggregative Feature Extraction

Andrea Schirru, Gian Antonio Susto, Simone Pampuri and Seán McLoone

Abstract— **Many modeling problems require to estimate a scalar output from one or more time series. Such problems are usually tackled by extracting a fixed number of features from the time series (like their statistical moments), with a consequent loss in information that leads to suboptimal predictive models. Moreover, feature extraction techniques usually make assumptions that are not met by real world settings (e.g. uniformly sampled time series of constant length), and fail to deliver a thorough methodology to deal with noisy data. In this paper a methodology based on functional learning is proposed to overcome the aforementioned problems; the proposed Supervised Aggregative Feature Extraction (SAFE) approach allows to derive continuous, smooth estimates of time series data (yielding aggregate local information), while simultaneously estimating a continuous shape function yielding optimal predictions. The SAFE paradigm enjoys several properties like closed form solution, incorporation of first and second order derivative information into the regressor matrix, interpretability of the generated functional predictor and the possibility to exploit Reproducing Kernel Hilbert Spaces setting to yield nonlinear predictive models. Simulation studies are provided to highlight the strengths of the new methodology w.r.t. standard unsupervised feature selection approaches.**

INTRODUCTION

Machine learning methodologies are nowadays applied in many industrial and scientific environments including technology-intensive manufacturing [5], biomedical sciences [6], and in general every data-intensive field that might benefit from reliable predictive capabilities. Machine learning techniques exploit organized data to create mathematical representations (*models*) of an observable phenomenon. It is then possible to rely on such a model to provide predictions for unobserved data. In mathematical terms, let

$$\mathcal{S} = \left\{ x_i \in \mathbb{R}^{1\times p},\ y_i \in \mathbb{R} \right\}_{i=1}^{N} \tag{1}$$

be a *training dataset* of N observations of a certain phenomenon. The i-th observation is characterized by p input features, constituting the vector x_i, and a scalar target value y_i. In practical terms, the input space usually relates to easily obtained data, while the target value is either not always available or results from a costly procedure; in a typical industrial application, x_i would collect sensor readings during a process operation, while y_i would be a quantitative indicator of product quality. The goal is then to exploit the information provided by $\mathcal{S}$ to create a predictive model f such that, given a new observation $\tilde{x} \notin \mathcal{S}$, $f(\tilde{x})$ will provide an accurate prediction of the unobserved $\tilde{y}$: in the case of the above mentioned industrial example, the model f would be able to estimate the final product quality relying only on sensor readings collected during process operation. It is important to note that in real life applications data is rarely (if ever) organized in a convenient $N\times p$ matrix ready to serve as input for a machine learning procedure. Indeed, the transition from a real life object to its mathematical representation will necessarily destroy part of the original information.

In this paper, we consider the learning problem where the input information is conveyed in the form of time series; more specifically, every observation of the phenomenon is described by p time series, that we know through an array of irregularly sampled measurements whose size can vary observation-wise. This setting relates to a common problem in predicting process results in an industrial setting [8], where the input space is often represented by non-uniformly sampled sensor readings. The challenge is to aggregate the information contained in each time series so that summary features are produced that are good predictors of the target value. Assuming the existence of a continuous process underlying such sensor readings, we adopt a functional learning paradigm in order to tackle the presented problem: specifically, we discuss suitable estimation techniques to reconstruct the original continuous time series and derive a feature extraction technique that can be employed with regular machine learning techniques, which we refer to as Supervised Aggregative Feature Extraction (SAFE). Furthermore, we prove the advantage of the proposed methodology w.r.t. other approaches by means of numerical simulations. The remainder of the paper is organized as follows: Section I provides a mathematical formalization of the problem at hand, as well as an overview of some feature extraction techniques for time series. Section II presents and discusses the proposed SAFE methodology for time series feature extraction, while Section III is devoted to underpinning basis expansion strategies and nonlinear regression techniques. Finally, Section IV validates the proposed methodology by means of numerical simulations. After the final remarks, Appendix A is devoted to mathematical proofs.

I. PROBLEM STATEMENT

Given N observations consisting of p time series, where the i-th observation $\mathcal{X}_i$ is defined as

$$\mathcal{X}_i = [x_i^{(1)}(t)\ \ldots\ x_i^{(j)}(t)\ \ldots\ x_i^{(p)}(t)],\ t \in [0,1], \forall\ j$$

A. Schirru and S. Pampuri are with University of Pavia, Italy. E-mail: {andrea.schirru, simone.pampuri}@unipv.it

G.A. Susto is with University of Padova, Italy. E-mail: gianantonio.susto@dei.unipd.it

S. McLoone is with National University of Ireland, Maynooth, Ireland. E-mail: sean.mcloone@eeng.nuim.ie

The financial support of the Irish Centre for Manufacturing Research and Enterprise Ireland (grant CC/2010/1001) are gratefully acknowledged.

978-1-4673-2066-5/12/$31.00 ©2012 IEEE

and a scalar target variable y_i, let the training set be

$$\mathcal{S} = \{\mathcal{X}_i,\ y_i\}_{i=1}^{N}$$

The goal is then to learn, relying on $\mathcal{S}$, a predictor function f. Such a predictor must be optimal in the sense that, given a new input $\mathcal{X}_{new}$, $f(\mathcal{S}, \mathcal{X}_{new})$ will be close (in the sense of a normed distance) to the unobserved y_{new}.

In practice, the continuous time series $x_i^{(j)}(t)$ are not available: instead it is necessary to rely on a set of discrete samples $\left\{t_{i,s}^{(j)},\ z_{i,s}^{(j)}\right\}_{s=1}^{\mathcal{N}_{i,j}}$ where $t_{i,s}^{(j)}$ and $z_{i,s}^{(j)}$ are the time and value of the s-th sampled point from the j-th time series of the i-th observation. In general, the series may have difform length (such that $\mathcal{N}_{i,j} \neq \mathcal{N}_{i,m},\ \mathcal{N}_{i,j} \neq \mathcal{N}_{k,j}$) and sampling timestamps ($t_{i,s}^{(j)} \neq t_{i,s}^{(m)}, t_{i,s}^{(j)} \neq t_{k,s}^{(j)}$). Furthermore, the noise of the channel needs to be taken into account:

$$\begin{aligned} z_{i,s}^{(j)} &= x_i^{(j)}(t_{i,s}^{(j)}) + v_{i,s}^{(j)} \\ v_{i,s}^{(j)} &\sim N(0, \rho_j^2) \end{aligned}$$

In order to employ machine learning techniques to find f, two main issues must be addressed: **(i)** it is in general necessary to extract a homogeneous set of features from every observation, and **(ii)** it is not possible to know in advance what part of the time series (if any) has an impact on the target variable. This lack of information must be taken in account when choosing a feature extraction methodology: indeed, a representation based solely on the global features of a dataset is likely to yield suboptimal predictions. In the next subsection, some of the most common feature extraction techniques for time series are presented and discussed.

A. Feature extraction

The extraction of a set of features from an observation will result in the loss of some information, especially when the format of such information is expected to show inter-example differences, such as in the presented case where difform sampling times and length are present. The goal is to build a regressor matrix $\Phi \in \mathbb{R}^{n \times \overline{p}}$, whose entry (i, j) represents the j-th feature of the i-th observation that can be subsequently used, along with the target variable vector $Y \in \mathbb{R}^n$, to train a predictor using a machine learning algorithm.

One of the simplest approaches is to rely on statistical moments: given p time series, let us build Φ as

$$\Phi = [\Phi_1\ \ldots\ \Phi_j\ \ldots \Phi_p]$$

where the $[i, k]$ element of $\Phi_j \in \mathbb{R}^{n \times k_{max}}$ is

$$\Phi_j[i, k] = m^{(k)}\left(\left\{z_{i,s}^{(j)}\right\}_{s=1}^{\mathcal{N}_{i,j}}\right)$$

Here k_{max} is the highest considered moment order and $m^{(k)}(\cdot)$ is the k-th sample moment of the input time series. It is immediately evident that this approach suffers from a major drawback, namely the inability to consider the dependency between information and time. Furthermore, it should be noted that the sample estimators of statistical moments are consistent for *independent* data points: it follows that, in the quite common case of autocorrelated time series, such estimates bear very little statistical meaning.

A more sophisticated approach consists of a systematic sampling of the input time series: specifically, the interval $[0, 1]$ is divided into $\mathcal{N}$ segments $[\tau_1\ \ldots\ \tau_{\mathcal{N}}]$. The regressor matrix is then populated with the segment-wise averages, as

$$\Phi_j[i, k] = \mathrm{Avg}[z_{i,s}^{(j)} : t_{i,s}^{(j)} \in \tau_k].$$

When using this approach it is necessary to select the number of segments, $\mathcal{N}$, in advance: this usually translates to a trade-off decision between locality (temporal resolution) and stability of information (robustness to noise). Furthermore, in the case of difform sampling, different features are likely to be computed from a different number of values, as the distribution of sampling points would privilege some segments over the others: this can potentially lead to data reliability issues. In order to overcome such instabilities, it is possible to project the rows of the Φ matrix obtained using the sampling approach on their direction of main variance. This yields the Principal Component Analysis (PCA) [4] transformation of the sampled input space.

B. Elements of machine learning and regularization

Once Φ is obtained, it is possible to employ a machine learning technique to find a predictor model f. As a first assumption, let the structure of the model be specified by a vector of parameters θ. Consider the fitness function

$$\mathcal{L}(\theta) = \mathcal{F}(\theta) + \lambda \mathcal{R}(\theta) \tag{2}$$

and the solution of the optimization problem

$$\theta^* = \arg\min_{\theta}\ \mathcal{L}(\theta).$$

The *error term* $\mathcal{F}$ measures the approximation power of f (w.r.t. $\mathcal{S}$), while the *regularization term* $\mathcal{R}$ measures the complexity of the model. Furthermore, $\lambda \geq 0$ is a *hyperparameter* that acts as a tuning knob for the trade-off between approximation and variability: too small a value results in an overfitted model (specifically tuned on the training set, with low predictive power), while too large a value results in an underfitted model (which would not incorporate the necessary information for making good predictions). The insight is that the correct value of λ would result in only the relevant information being incorporated into the model, yielding the highest predictive power.

While a wide variety of choices are possible for both $\mathcal{F}$ and $\mathcal{R}$, most learning techniques require the minimization problem to be convex w.r.t. θ; to exploit this desirable feature, it is sufficient for $\mathcal{F}$ and $\mathcal{R}$ to be convex. Notably, if f is defined as a linear function of θ such as

$$f(\Phi; \theta) := \Phi\theta$$

and $\mathcal{F}$ is the sum of squared estimation residuals

$$\mathcal{F} := ||Y - f(\Phi; \theta)||^2 = ||Y - \Phi\theta||^2$$

the global minimum of $\mathcal{F}$ can be expressed in closed form as the *least squares* solution

$$\theta^* = (\Phi'\Phi)^{-1}\Phi' Y \tag{3}$$

Equation (3) is prone to numerical issues and instability, since there is no guarantee that $\Phi'\Phi$ will be full rank or well conditioned. A modification that preserves this closed-form solution and resolves instability issues is represented by Ridge Regression (RR), obtained by setting

$$\mathcal{R}(\theta) := \theta'\theta = \sum_i \theta_i^2.$$

The global RR minimizer is then

$$\theta^* = (\Phi'\Phi + \lambda I)^{-1}\Phi' Y \tag{4}$$

In order to obtain a nonlinear model f without giving up the desirable convexity features of the optimization problem, it is possible to exploit the *kernel trick* [1] to embed a nonlinear projection of Φ on a Reproducing Kernel Hilbert Space (RKHS) [2] in a quadratic optimization problem. Noting that

$$(\Phi'\Phi + \lambda I)^{-1}\Phi' = \Phi'(\Phi\Phi' + \lambda I)^{-1}$$

equation (4) may be rewritten as

$$\theta^* = \Phi'(\Phi\Phi' + \lambda I)^{-1} Y$$

and the prediction $f(\Phi_{new})$ as

$$f(\Phi_{new}) = \langle \Phi_{new}, \Phi \rangle \left(\langle \Phi, \Phi \rangle + \lambda I\right)^{-1} Y \tag{5}$$

By replacing the linear inner product $\langle \cdot, \cdot \rangle$ with a nonlinear positive definite kernel function $\mathcal{K}$, the *Kernel Ridge Regression* [10] coefficient vector c can be defined as

$$c^* = (\mathcal{K}(\Phi, \Phi) + \lambda I)^{-1} Y$$

and the corresponding predictor f is given by

$$f(\Phi_{new}) = \mathcal{K}(\Phi_{new}, \Phi) c^*$$

Thus, the resulting model yields nonlinear predictions w.r.t. the elements of the regressor matrix. A thorough review of machine learning techniques and Kernel-based techniques is beyond the scope of this paper. The interested reader is referred to [4] and [9].

II. Supervised aggregative feature extraction

In this section the proposed supervised aggregative feature extraction (SAFE) methodology is presented and motivated from a theoretical point of view. In order to introduce SAFE, we consider an ideal case, in which the continuous functions $x_i^{(j)}(t)$ are known and available. Employing the functional regression paradigm, consider the following definition of f:

$$f(\mathcal{X}_i) := \sum_{j=1}^{p} \left\langle x_i^{(j)}(t), \beta^{(j)}(t) \right\rangle_{L^2} \tag{6}$$

where $\langle f, g \rangle_{L^2}$ is the L^2 inner product of real functions f and g, defined as

$$\langle f, g \rangle_{L^2} = \int_{-\infty}^{\infty} f(t) g(t) dt$$

It is apparent how the predictor defined by (6) assumes that the continuous phenomenon x influences the target variable y through a weighted integration with an unknown shape function β. In the following we focus on the sum of squared residuals approximation error term, defined as

$$\mathcal{F}(\beta) = \sum_{i=1}^{N} \left(\sum_{j=1}^{p} \int_{-\infty}^{\infty} \beta^{(j)}(t) x_i^{(j)}(t) dt - y_i \right)^2 \tag{7}$$

It is then possible to introduce the functional learning optimization problem:

$$\beta^* = \arg\min_{\beta} \mathcal{F}(\beta) + \lambda \mathcal{R}(\beta) \tag{8}$$

$$\beta = \left[\beta^{(1)}(t),\ \beta^{(j)}(t),\ \beta^{(p)}(t) \right] \tag{9}$$

where $\mathcal{F}(\beta)$ is defined in (7) and $\mathcal{R}(\beta)$ is a regularization term that penalizes the variability of β: for example,

$$\mathcal{R}(\beta) = \sum_{j=1}^{p} \left\langle \beta^{(j)}, \beta^{(j)} \right\rangle_{L^2}$$

It is apparent that the shape functions $\beta^{(\cdot)}(t)$ are functional parameters of the optimization problem (9). It is to be noted that it is not possible to directly handle (7) for two reasons: **(i)** the functions $x_i^{(j)}(t)$ are observed only through a finite number of noisy, irregularly sampled data points; **(ii)** the generic functions $\beta^{(j)}(t)$ have infinite degrees of freedom. To overcome such issues and solve (9), the next sections present a Gaussian process estimation of the unobserved time series and propose a parametrization for the shape functions β.

A. Time series approximation

Consider an approximation of the fitness function $\mathcal{L}$

$$\hat{\mathcal{L}} = \hat{\mathcal{F}} + \lambda \mathcal{R}$$

where the approximated loss function is defined as

$$\hat{\mathcal{F}} = \sum_{i=1}^{N} \left(\sum_{j=1}^{p} \int_{-\infty}^{\infty} \beta^{(j)}(t) \hat{x}_i^{(j)}(t) dt - y_i \right)^2$$

and $\hat{x}_i^{(j)}(t)$ is an estimate of the unobserved $x_i^{(j)}(t)$. In order to obtain this estimate we consider the expected value of a monodimensional Gaussian process posterior distribution. According to Riesz's representation theorem [7] a continuous interpolation of $x_i^{(j)}(t)$ from its samples is given by

$$\hat{x}_i^{(j)}(t) = \sum_{s=1}^{\mathcal{N}_{i,j}} \mathcal{K}(t, t_{i,s}^{(j)}) c_{i,s}^{(j)}$$

where $\mathcal{K}$ is a suitable positive definite kernel function. The vector $c_{i,\cdot}^{(j)}$ is obtained as

$$c_{i,\cdot}^{(j)} = (\mathbf{K} + \xi_j I)^{-1} x_{i,\cdot}^{(j)}$$

where the $[w, z]$ entry of the kernel matrix $\mathbf{K}$ is

$$\mathbf{K}_{[w,z]} = \mathcal{K}\left(t_{i,w}^{(j)}, t_{i,z}^{(j)}\right)$$

and $x_{i,\cdot}^{(j)}$ is the column vector of the available observations. It is immediately evident how every coefficient of $c_{i,\cdot}^{(j)} \in \mathcal{N}_{i,j}$

depends on all the observed points. Considering the radial basis function kernel and the Gaussian density, such that

$$\mathcal{K}(t_1,t_2) := e^{-\frac{(t_1-t_2)^2}{2\omega^2}} \quad (10)$$

$$G(a,b;x) := \frac{1}{\sqrt{2\pi}b}e^{-\frac{(a-x)^2}{2b^2}} \quad (11)$$

it follows that

$$\begin{aligned}\mathcal{K}(t_1,t_2) &= \sqrt{2\pi}\omega G(t_1,\omega^2;t_2)\\ \hat{x}_i^{(j)}(t) &= \sqrt{2\pi}\omega_{(j)}\sum_{s=1}^{\mathcal{N}_{i,j}} c_{i,s}^{(j)} G(t_{i,s}^{(j)},\omega_{(j)}^2;t). \quad (12)\end{aligned}$$

Hence, the continuous-time approximation of $x_i^{(j)}(t)$ is obtained as a weighted sum of Gaussian densities. It should be noted that, to obtain such approximation, it is necessary to select two hyperparameters for each time series, namely the regularization term ξ_j and the kernel bandwdith $\omega_{(j)}^2$.

B. Shape function parametrization

Let us consider a linear combination of Gaussian densities as parametrization for $\beta^{(j)}$, such that

$$\begin{aligned}\beta^{(j)}(t) &= \sum_{k=1}^{\gamma}\alpha_k^{(j)}G(\mu(k),\sigma^2;t)\\ \mu(k) &= \frac{k-1}{\gamma-1}\end{aligned}$$

where the parameter γ controls the number of base Gaussian components, and σ^2 is the bandwidth of the Gaussian density. The approximate loss function $\hat{\mathcal{F}}$ takes the following form:

$$\begin{aligned}\hat{\mathcal{F}} &= \sum_{i=1}^{N}\left(\sum_{j=1}^{p}\int_{-\infty}^{\infty}\left(\sum_{k=1}^{\gamma}\alpha_k^{(j)}G(\mu(k),\sigma^2;t)\times\right.\right.\\ &\quad\left.\left.\times\sum_{s=1}^{\mathcal{N}_{i,j}}\sqrt{2\pi}\omega_{(j)}G(t_{i,s}^{(j)},\omega_{(j)}^2;t)c_{i,s}^{(j)}\right)dt - y_i\right)^2\\ &= \sum_{i=1}^{N}\left(\sqrt{2\pi}\sum_{j=1}^{p}\omega_{(j)}\sum_{k=1}^{\gamma}\alpha_k^{(j)}\sum_{s=1}^{\mathcal{N}_{i,j}}c_{i,s}^{(j)}\times\right.\\ &\quad\left.\times\int_{-\infty}^{\infty}\left(G(\mu(k),\sigma^2;t)G(t_{i,s}^{(j)},\omega_{(j)}^2;t)\right)dt - y_i\right)^2\end{aligned}$$

Considering the following Theorem (proof in Appendix A)

Theorem 2.1: Let $a,b,x\in\mathbb{R}^p$ and $A,B\in\mathbb{R}^{p\times p}$. It holds that $\int_{-\infty}^{\infty}G(a,A;x)G(b,B;x)dx = G(a,A+B;b)$ where G is the Gaussian density as in (11).

allows $\hat{\mathcal{F}}$ to be rewritten as

$$\hat{\mathcal{F}} = \sum_{i=1}^{N}\left(\sqrt{2\pi}\sum_{j=1}^{p}\omega_{(j)}\sum_{k=1}^{\gamma}\alpha_k^{(j)}\times\right.$$
$$\left.\times\sum_{s=1}^{\mathcal{N}_{i,j}}c_{i,s}^{(j)}G(\mu(k),\sigma^2+\omega_{(j)}^2;t_{i,s}^{(j)})-y_i\right)^2.$$

Defining the parameters

$$\delta_{i,s}^{(j)}(k) = \sqrt{2\pi}c_{i,s}^{(j)}\omega_j G(\mu(k),\sigma^2+\omega_{(j)}^2;t_{i,s}^{(j)}) \quad (13)$$

$$\overline{\delta}_i^{(j)}(k) = \sum_{s=1}^{\mathcal{N}_{i,j}}\delta_{i,s}^{(j)}(k), \quad (14)$$

yields the compact version of $\hat{\mathcal{F}}$ as

$$\hat{\mathcal{F}} = \sum_{i=1}^{N}\left(\sum_{j=1}^{p}\sum_{k=1}^{\gamma}\alpha_k^{(j)}\overline{\delta}_i^{(j)}(k)-y_i\right)^2 = \hat{\mathcal{F}} = ||\Phi\theta - Y||^2$$

with $\Phi=\Delta$, where

$$\Delta = \begin{bmatrix}\overline{\delta}_1^{(1)}(1) & \dots & \overline{\delta}_1^{(1)}(\gamma) & \overline{\delta}_1^{(2)}(1) & \dots & \overline{\delta}_1^{(p)}(\gamma)\\ \vdots & & \vdots & \vdots & & \vdots\\ \overline{\delta}_N^{(1)}(1) & \dots & \overline{\delta}_N^{(1)}(\gamma) & \overline{\delta}_N^{(2)}(1) & \dots & \overline{\delta}_N^{(p)}(\gamma)\end{bmatrix}$$

$$\theta = \begin{bmatrix}\alpha_1^{(1)} & \alpha_2^{(1)} & \dots & \alpha_k^{(j)} & \dots & \alpha_\gamma^{(p)}\end{bmatrix}^T$$

and Y is the vector of output observations. Since $\hat{\mathcal{F}}$ is a quadratic form of the coefficients α, it is convex. If $\mathcal{R}$ is convex as well, the solution of the problem can be found by solving $\frac{\partial\hat{\mathcal{L}}}{\partial\theta}=0$ w.r.t. θ. For instance, the RR solution follows from (4).

III. Derivatives basis expansion

In this section, the convenient properties of the proposed approximation (12) are exploited to expand the regressors matrix to include information about its derivatives. The theory behind first- and second-order derivative expansion is covered and the corrisponding formulae are provided. Let us consider the first derivative of $\hat{x}_i^{(j)}(t)$

$$\frac{\partial\hat{x}_i^{(j)}(t)}{\partial t} = -\frac{\sqrt{2\pi}}{\omega_{(j)}}\sum_{s=1}^{\mathcal{N}_{i,j}}G(t_{i,s}^{(j)},\omega_{(j)}^2;t)c_{i,s}^{(j)}(t-t_{i,s}^{(j)})$$

By exploiting the following

Theorem 3.1: Letting all the quantities be as in Theorem 2.1, it holds that $\int_{-\infty}^{\infty}G(a,A;x)\frac{\partial G(b,B;x)}{\partial x}dx = \Omega G(a,A+B;b)$ with $\Omega=\left(\frac{b-a}{A+B}\right)$

it is possible to define

$$\tau_{i,s}^{(j)}(k) = -\left(\frac{\delta_{i,s}^{(j)}(k)}{\omega_{(j)}^2}\right)\left(\frac{\mu(k)-t_{i,s}^{(j)}}{\sigma^2+\omega_{(j)}^2}\right) \quad (15)$$

$$\overline{\tau}_i^{(j)}(k) = \sum_{s=1}^{\mathcal{N}_{i,j}}\tau_{i,s}^{(j)}(k) \quad (16)$$

and use the matrix T, whose elements are $T[j,k]=\overline{\tau}_i^{(j)}(k)$, as a basis expansion for Δ, such that $\Phi=[\Delta\ T]$. Similarly, the second derivative of $\hat{x}_i^{(j)}(t)$ is

$$\frac{\partial^2\hat{x}_i^{(j)}(t)}{\partial^2 t} = \frac{\sqrt{2\pi}}{\omega_{(j)}^3}\sum_{s=1}^{\mathcal{N}_{i,j}}G(t_{i,s}^{(j)},\omega_{(j)}^2;t)c_{i,s}^{(j)}((t-t_{i,s}^{(j)})^2-\omega_{(j)}^2)$$

and, using the following

Theorem 3.2: Letting all the quantities be as in Theorem 2.1, it holds that $\int_{-\infty}^{\infty} G(a,A;x)\frac{\partial^2 G(b,B;x)}{\partial^2 x}dx = \Gamma G(a,A+B;b)$ with

$$\Gamma = \frac{(a-b)^2 - (A+B)}{(A+B)^2} = \Omega^2 - \frac{1}{A+B}$$

where Ω is as defined in Theorem 3.1.

the second derivative basis expansion elements read

$$\eta_{i,s}^{(j)}(k) = \left(\frac{\delta_{i,s}^{(j)}(k)}{\omega_{(j)}^4}\right)\left(\frac{(\mu(k)-t_{i,s}^{(j)})^2 - (\sigma^2+\omega_{(j)}^2)}{(\sigma^2+\omega_{(j)}^2)^2}\right)$$

$$\overline{\eta}_i^{(j)}(k) = \sum_{s=1}^{\mathcal{N}_{i,j}} \eta_{i,s}^{(j)}(k) \quad (17)$$

The matrix H of the elements $\overline{\eta}_i^{(j)}(k)$ is then similarly used to expand the matrix Φ, as $\Phi = [\Delta\ T\ H]$.

IV. Experimental results

A. Experimental setup

The proposed methodology was tested against the feature extraction techniques defined in Section I, namely *Statistical moments*, *Systematic sampling* and *PCA*. The input matrices resulting from such methodologies are employed to build an optimal RR model; 500 instances of every synthetic dataset were created, each one composed of a training set and a test set (100 and 50 examples). Each example consists of a single input time series (available through a number of sampling points uniformly distributed between 35 and 45) and an output target value. Gaussian distributed white noise $\mathcal{N}(0, 0.1)$ was imposed on every sampled time series value and on every target value. The methodologies were evaluated using the Root Mean Squared Error (RMSE) on the test data as a performance metric. For every experiment, the SAFE technique was tested with and without the inclusion of the time series first and second derivative expansions.

B. The sinusoid dataset

The purpose of the sinusoid dataset is to reproduce a situation in which only an unknown part of the input time series influence the target variable. In mathematical terms, the input time series is defined as follows:

$$x(t) = \sin(t\omega + \delta)$$
$$\omega \sim \mathcal{U}(0.01, 10) \qquad \delta \sim \mathcal{U}(0, 2\pi)$$

while the target variable is computed as

$$y = \int_{0.3}^{0.7} x(t)dt = \frac{\cos(0.3\omega+\delta) - cos(0.7\omega+\delta)}{\omega}$$

Figure 1 shows the results for the sinusoid dataset: it is apparent that, while the statistical moment-based feature extraction is not able to learn a correct model, all the other techniques yield almost the same performances. This is quite unsurprising, since the statistical moment extraction relies exclusively on global features, and is therefore unable to select the correct range in the input time series. By analytically inspecting the RMSE results the SAFE methodology yields marginally better results w.r.t. sampling- and PCA-based feature extraction.

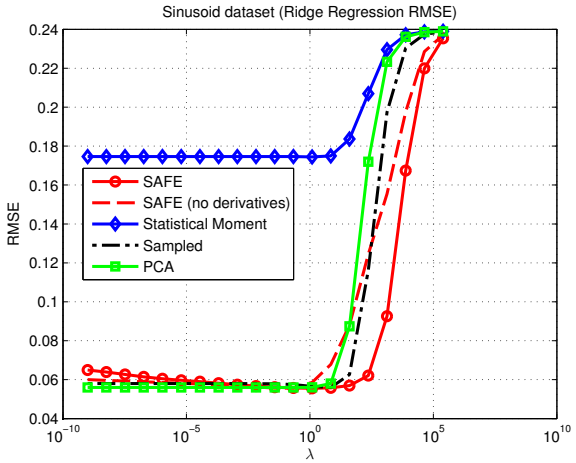

Fig. 1. Sinusoid dataset results (average over 500 simulations)

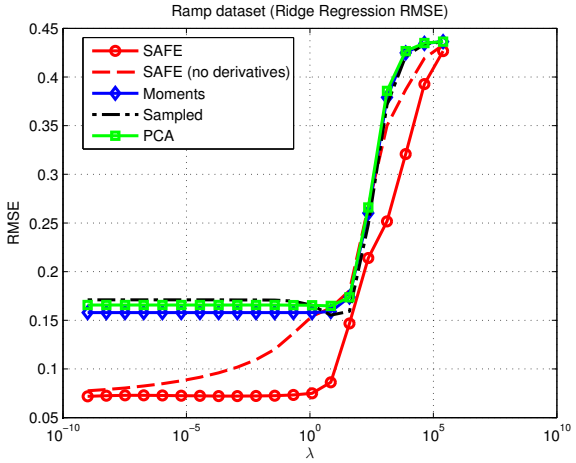

Fig. 2. Ramp dataset results (average over 500 simulations)

C. The ramp dataset

The goal of the ramp dataset is to highlight the advantages of including time series derivative information in the extracted features. The input time series is generated as

$$x(t) = \begin{cases} n_1\sqrt{2t} & t < 0.5 \\ n_1 + n_2(t-0.5) & t \geq 0.5 \end{cases}$$
$$n_1 \sim \mathcal{U}(0,1), \quad n_2 \sim \mathcal{U}(1,4),$$

while the output variable reads

$$y = n_2.$$

In other words, the slope of the second part of $x(t)$ (for $t \geq 0.5$) is the target variable. Figure 2 shows the test results for the ramp dataset. As expected, the incorporation of derivative information in the input dataset allows expanded SAFE to outperform the other methodologies.

CONCLUSIONS

In this paper, a novel feature extraction framework is presented for dataset consisting of time series input spaces and scalar target variable. The research is originally motivated by real-life datasets representing industrial processes, but the presented results are appliable to any time series-intensive learning environment (such as bioengineering). The proposed methodology, SAFE, derives from a functional learning setting in which the time series input space is reconstructed by means of Gaussian process inference, and the unknown shape function is parametrized as a weighted sum of Gaussian functions. This setup allows for a number of interesting properties, including closed form solution and the possibility of using the extracted information as input for any machine learning methodology. The capabilities of the SAFE methodology have been assessed by means of simulated examples, with the purpose of testing the novel framework against similar techniques. Such benchmarks yield promising preliminary results, as the proposed methodology is able to obtain in general better results than its competitors, including situations where the target output is determined by global features of the input time series.

APPENDIX A

Let $G(b, B; x)$ be the univariate Gaussian probability distribution function of expected value b and variance B as in (11). By applying derivative rules, it follows that

$$\frac{\partial G(b,B;x)}{\partial x} = -\left(\frac{x-b}{B}\right) G(b,B;x) \quad (18)$$

$$\frac{\partial^2 G(b,B;x)}{\partial^2 x} = \frac{G(b,B;x)}{B^2}((x-b)^2 - B) \quad (19)$$

In the following we consider the theorem proposed in [11] in the special case for which $s = t = 1$ and $\mathbf{Q} = 1$.

Theorem 4.1: Let $\mathbf{A} \in \mathbb{R}^{s\times s}$, $\mathbf{a} \in \mathbb{R}^s$, $\mathbf{B} \in \mathbb{R}^{t\times t}$, $\mathbf{b} \in \mathbb{R}^t$ and $\mathbf{Q} \in \mathbb{R}^{s\times t}$. Let $\mathbf{x} \in \mathbb{R}^t$ be an input variable. It holds that

$$G(\mathbf{a}, \mathbf{A}; \mathbf{Q}\mathbf{x}) G(\mathbf{b}, \mathbf{B}; \mathbf{x}) = G(\mathbf{a}, \mathbf{A} + \mathbf{Q}\mathbf{B}\mathbf{Q}'; \mathbf{b}) \times \times G(\mathbf{d}, \mathbf{D}; \mathbf{x})$$

with $\mathbf{D} = (\mathbf{Q}'\mathbf{A}^{-1}\mathbf{Q} + \mathbf{B}^{-1})^{-1}$ and $\mathbf{d} = \mathbf{b} + \mathbf{D}\mathbf{Q}'\mathbf{A}^{-1}(\mathbf{a} - \mathbf{Q}\mathbf{b})$

Proof of Theorem 2.1. Let

$$\chi = \int_{-\infty}^{\infty} G(a, A; x) G(b, B; x) dx$$

By applying Theorem 4.1,

$$\chi = G(a, A + B; b) \int_{-\infty}^{\infty} G(d, D; x) dx \quad (20)$$

Since by definition $\int_{-\infty}^{\infty} G(d, D; x) dx = 1$ it holds that $\chi = G(a, A + B; b)$. □

Proof of Theorem 3.1

$$\begin{aligned}
\chi &= \int_{-\infty}^{\infty} G(a, A; x) \frac{\partial G(b, B; x)}{\partial x} dx \\
&= -\frac{1}{B} \int_{-\infty}^{\infty} (x - b)\, G(a, A; x) G(b, B; x) \\
&= -\frac{G(a, A + B; b)}{B} \int_{-\infty}^{\infty} (x - b)\, G(d, D; x) dx \\
&= -\frac{G(a, A + B; b)}{B} (d - b)
\end{aligned}$$

Since, following Theorem 4.1,

$$d - b = DA^{-1}(a - b) = \frac{A^{-1}}{A^{-1} + B^{-1}}(a - b)$$

it holds that $-\frac{1}{B}\frac{A^{-1}}{A^{-1}+B^{-1}}(a - b) = -\frac{a-b}{A+B}$ and therefore $\chi = -\left(\frac{a-b}{A+B}\right) G(a, A + B; b)$. □

Proof of Theorem 3.2.

$$\begin{aligned}
\chi &= \int_{-\infty}^{\infty} G(a, A; x) \frac{\partial^2 G(b, B; x)}{\partial^2 x} dx \\
&= \frac{1}{B^2} \int_{-\infty}^{\infty} G(a, A; x) G(b, B; x) \left((x - b)^2 - B\right) dx \\
&= \frac{G(a, A + B; b)}{B^2} \int_{-\infty}^{\infty} G(d, D; x) \left((x - b)^2 - B\right) dx
\end{aligned}$$

Since $(x - b)^2 = (x - d)^2 + b^2 - d^2 - 2bx + 2dx$ and $\int_{-\infty}^{\infty} G(d, D; x)(x - d)^2 dx = D$ it holds that

$$\int_{-\infty}^{\infty} G(d, D; x)((x - b)^2 - B) dx = D + (b - d)^2 - B$$

and therefore

$$\begin{aligned}
\chi &= \frac{D + (b - d)^2 - B}{B^2} G(a, A + B; b) \\
&= \frac{(a - b)^2 - (A + B)}{(A + B)^2} G(a, A + B; b).
\end{aligned}$$

□

REFERENCES

[1] A. Aizerman, E.M. Braverman, L.I. Rozoner *Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning*, Automation and Remote Control 25, 821-837 (1964).

[2] N. Aronszajn, *Theory of Reproducing Kernels*, Transactions of the American Mathematical Society 68(3), 337-404 (1950).

[3] H.-J. Dai, Y.-C. Chang, R.T.-H., Tsai, W.-L. Hsu *New Challenges for Biological Text-Mining in the Next Decade*, Journal of Computer Science and Technology 25(1), 169-179 (2010).

[4] T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning. Data Mining, Inference and Prediction.*, Springer (2009).

[5] L. Monostori *AI and machine learning techniques for managing complexity, changes and uncertainties in manufacturing*, Engineering Applications of Artificial Intelligence 16(4), 277-291 (2003).

[6] G. Pillonetto, F. Dinuzzo, G. De Nicolao, *Bayesian Online Multitask Learning of Gaussian Processes*, IEEE Transactions on Pattern Analysis and Machine Intelligence 32(2), 193-205 (2010).

[7] W. Rudin, *Real and Complex Analysis*, McGraw-Hill (1966).

[8] A. Schirru, S. Pampuri, C. De Luca, G. De Nicolao, *Multilevel Kernel Methods for Virtual Metrology in Semiconductor Manufacturing*, Proceedings of the 18th IFAC World Congress, Milan (2011).

[9] B. Scholkopf, A. Smola *Learning with Kernels*, The MIT Press (2001).

[10] A.N. Tikhonov, *On the Stability of Inverse Problems*, C.R. (Doklady) Acad. Sci. URSS (N.S.) 39, 176-179 (1943).

[11] K.S. Miller, *Multidimensional Gaussian distributions*, Wiley, 1964.