An Adaptive Online Learning Approach for Support Vector Regression: Online-SVR-FID

Jie LIU[1] and Enrico ZIO[1,2,*]

[1] Chair on Systems Science and the Energetic Challenge, Foundation Électricité de France (EDF), CentraleSupélec, Grande Voie des Vignes, 92295 Châtenay-Malabry, France

[2] Energy Departement, Politecnico di Milano, Campus Bovisa,Via Lambruschini, 4, 20156, Milano, Italy

[*]Corresponding author

Email: jie.liu@centralesupelec.fr; enrico.zio@centralesupelec.fr

Abstract

Support Vector Regression (SVR) is a popular supervised data-driven approach for building empirical models from available data. Like all data-driven methods, under non-stationary environmental and operational conditions it needs to be provided with adaptive learning capabilities, which might become computationally burdensome with large datasets cumulating dynamically. In this paper, a cost-efficient online adaptive learning approach is proposed for SVR by combining Feature Vector Selection (FVS) and Incremental & Decremental Learning. The proposed approach adaptively modifies the model only when different pattern drifts are detected according to proposed criteria. Two tolerance parameters are introduced in the approach to control the computational complexity, reduce the influence of the intrinsic noise in the data and avoid the overfitting problem of SVR. Comparisons of the prediction results is made with other online learning approaches e.g. NORMA, SOGA, KRLS, Incremental Learning, on several artificial datasets and a real case study concerning time series prediction based on data recorded on a component of a nuclear power generation system. The performance indicators MSE and MARE computed on the test dataset demonstrate the efficiency of the proposed online learning method.

Key words:
Online learning, Support vector regression, Time series data, Pattern drift, Feature vector selection, Incremental & Decremental learning

1. Introduction

Various efforts of research on machine learning have been devoted to studying situations in which a sufficiently large and representative dataset is available from a fixed, albeit unknown distribution. The models trained for these situations can function well only for patterns within the representative training dataset [20].

In real-world applications, systems/components are usually operated in non-stationary environments and evolving operational conditions, whereby patterns drift. Then, to be of practical use the models built must be capable of timely learning changes in the existing patterns and new

patterns arising in the dynamic environment of system/component operation. The up-to-date snapshot of the ongoing research in this area can be found in [6], [7], [13], [20], [21], [22]. Different adaptive strategies have been proposed for neural networks [6], Markov Chain [7], fuzzy inference systems [21], feature extraction [13], and fault detection [22].

Support Vector Regression (SVR) is a popular, supervised data-driven approach, which also must cope with the problem of changing environments and adaptation to pattern drifts, but has attracted relatively less attention for adaptive learning. This paper provides a strategy for feature extraction and adaptive learning with SVR.

Some approaches have been proposed in the literature for SVR to adaptively learn new data points [2], [3], [8], [10], [18], [23]. In these approaches, the online learning of a trained SVR model is mostly guided by prediction accuracy and/or characteristics of the inputs of the data points: increasing data points are used to update the SVR model when the corresponding predictions are not precise and/or they belong to a less explored zone of the input space and, thus, contain new information. With respect to the consideration of the input characteristics for achieving model update, reference [18] proposes an approach based on an adaptive Kernel Principal Component Analysis (KPCA) and Support Vector Machine (SVM) for real-time fault diagnosis of High-Voltage Circuit Breakers (HVCBs). Bordes et al. [2] propose an online algorithm which converges to the SVM solution by using the τ-violating pair paradigm. Wang et al. [23] propose an online core vector machine classifier with adaptive Minimum-Enclosing-Ball (MEB) adjustment. Reference [10] uses a small subset of basis vectors to approximate the full kernel on arbitrary points. Engel et al. [8] present a nonlinear kernel-based recursive least squares algorithm, which performs linear regression in the feature space and can be used to recursively construct the minimum mean squared-error regressor. Csato and Opper [3] combine a Bayesian online algorithm with a sequential construction of relevant subsets of the training dataset and propose Sparse On-line Gaussian Process (SOGP) to overcome the limitation of Gaussian process on large datasets.

The methods above consider only the characteristics of the inputs to update the model, not the prediction accuracy. Reference [4] proposes an online recursive algorithm to "adiabatically" add or remove one data point in the model while retaining the Kuhn-Tucker conditions on all the other data points in the model. Martin [17] further develops this method for the incremental addition of new data points, removal of existing points and update of target values for existing data points. But the authors provide only the "how" for model update, while the "when" and "where" to make such update are not presented, and adding each new point available can soon become quite time-consuming. Karasuyama and Takeuchi [11] propose a multiple incremental algorithm of SVM, based on the previous results. These above mentioned incremental and decremental learning approaches feed to the model all new points including noisy and useless ones, without bothering of selecting the most informative ones. Crammer et al. [5] propose online passive-aggressive algorithms for classification and regression, but considering only the prediction accuracy as the update criterion. Reference [12] considers using classical stochastic gradient descent within a feature space and some straightforward manipulations for online learning with kernels. The gradient descent method destroys completely the Kuhn-Tucker conditions, which instead are necessary for building a SVR model.

In this paper, the authors propose an online learning approach for SVR to adaptively modify the model when different types of pattern drifts are detected, providing a solution for "when" and "where" to modify the trained model. The proposed online learning approach is a compromise

among prediction accuracy, robustness and computational complexity, obtained by combining a simplified version of the Feature Vector Selection (FVS) method introduced in [1] with the Incremental & Decremental Learning presented in [4], considering the characteristics of the inputs of new data points and the bias of the corresponding prediction. The method is hereafter called Online learning approach for SVR using FVS and Incremental & Decremental Learning, Online-SVR-FID for short. FVS aims at reducing the size of the training dataset: instead of training the SVR model with the whole training dataset, only part of it (the set of Feature Vectors (FVs) which are nonlinearly independent in the Reproduced Kernel Hilbert Space (RKHS)) is used and the mapping of the other training data points in RKHS can be expressed by a linear combination of the selected FVs. In this paper, FVS is simplified and used for the proposed adaptive online learning approach. According to the geometric meaning of FVS in RKHS, in this paper, each data point (input-output pair) is defined as a pattern and two types of pattern drifts are given: new pattern and changed pattern. A new data point is a new pattern (or new FV) if the mapping of its inputs in RKHS cannot be represented by a linear combination of the mapping of existing patterns (this is integrated in some papers), while it is a changed pattern if its mapping can be represented by such a linear combination but the bias of its predicted value is bigger than a predefined threshold. Once a new data point is judged as a new pattern, it is immediately added to the present model no matter the bias of its prediction is small or big, thus keeping the richness of the patterns in the model. A changed pattern is used to replace a selected existing pattern instead of adding it into the model, thus keeping the nonlinear independence in RKHS among all the data points in the model, which is critical for FVS calculation. When adding or removing a FV in the model, instead of retraining the model, Incremental & Decremental Learning can construct the solution iteratively. Two criteria are proposed to detect new and changed patterns, considering respectively the characteristics of the inputs and bias of the prediction. The proposed approach can efficiently add new patterns and change existing patterns in the model, to follow the incoming patterns and at the same time reduce the computational burden by selecting only informative data points. The two criteria proposed for verification of new patterns and changed patterns can also help avoiding the overfitting problem bothering SVR and reducing the influence of the intrinsic noise in the data.

The structure of the proposed Online-SVR-FID is similar to that of the method proposed in [23]. However, the method proposed in [23] is based on MEB whereas Online-SVR-FID is based on FVS. Another main difference between these two methods is that the method proposed in [23] adds only the new patterns (as defined previously) in the model, while in Online-SVR-FID, two kinds of drifts (new patterns and changed patterns) are identified. New patterns are added to the model directly as in [23] whereas the changed patterns are used to replace existing FVs to keep the nonlinear independence among FVs, which is critical during online learning.

Four artificial datasets with pattern drifts are firstly tested to compare the results produced by Online-SVR-FID and some benchmark methods. A real case study is, then, worked out concerning the leak flow from a seal of a pump in a Nuclear Power Plant (NPP). The comparison with several other online learning methods proves the accuracy and efficiency of the proposed method.

The rest of the paper is organized as follows. Section 2 gives some basics of SVR, the modified FVS and Incremental & Decremental Learning; the proposed Online-SVR-FID is also detailed in this section. Section 3 describes the artificial and real case studies, and presents the experimental results and their comparisons with other online learning methods. Some conclusions and perspectives are drawn in Section 4.

2. Online-SVR-FID

Pattern drift is a challenging problem for supervised data-driven approaches. The Online-SVR-FID approach proposed in this paper is a cost-efficient online learning approach for SVR, capable of handling new patterns and changed patterns as defined in the Introduction. It can effectively and timely detect and add a new pattern or update a changed pattern in the model, while retaining the Kuhn-Tucker conditions, which are necessary and sufficient conditions for the optimization of the quadratic function associated to SVR. Two criteria considering the characteristics of the input and the bias of the prediction, are proposed for verification of the two types of pattern drifts.
In order to fully explore the Online-SVR-FID, we briefly recall SVR, FVS [1] and Incremental & Decremental Learning [4]. The proposed approach is, then, detailed, a pseudo-code is given and two tolerance parameters are introduced for computational control.

2.1 Support Vector Regression with ε-Insensitive Loss Function

SVR seeks to find the best estimate function $f(\boldsymbol{x}) = \boldsymbol{\omega x} + b$ of the real underlying function for a set of training data points $(\boldsymbol{x}_i, y_i)$, for $i = 1, 2, \ldots, T$. By solving the Kuhn-Tucker conditions of the following quadratic optimization problem

$$\text{Minimize } \frac{1}{2}\|\boldsymbol{\omega}\|^2 + C\sum_{i=1}^{T}(\xi_i + \xi_i^*)$$

$$\text{Subject to } \begin{cases} y_i - \boldsymbol{\omega x}_i - b \le \varepsilon + \xi_i \\ \boldsymbol{\omega x_i} + b - y_i \le \varepsilon + \xi_i^*, \\ \xi_i, \xi_i^* \ge 0 \end{cases} \quad (1)$$

the best estimate function $f(\boldsymbol{x})$ can be expressed as a support vector expansion

$$f(\boldsymbol{x}) = \sum_{i=1}^{T}\beta_i k(\boldsymbol{x}, \boldsymbol{x}_i) + b, \quad (2)$$

where $k(\boldsymbol{x}, \boldsymbol{x}_i) = e^{-\|\boldsymbol{x}-\boldsymbol{x}_i\|^2/2\sigma^2}$ in the case of Radial Basis Function (RBF); the multipliers (also called influences in some literature) $\beta_i \in [-C, C]$, for $i = 1, \ldots, T$ are the solutions of the dual optimization problem in SVR and satisfy the corresponding Kuhn-Tucker conditions. Details can be found in [9]. The points $\boldsymbol{x}_i$ with non-zero multipliers β_i are called Support Vectors (SVs).
There are three hyperparameters in the SVR model using RBF kernel function and the ε-insensitive loss function: the penalty factor *C*, the sparsity of the data ε and the width of the kernel σ.

2.2 Feature vector selection

Baudat and Anouar [1] define two parameters (local fitness and global fitness) to characterize the feature space of training dataset. A number of FVs are selected from the mapping of all training data points to represent the useful dimension of RKHS in the training dataset. Mapping of any other data points in RKHS can be projected on these FVs and, then, classical algorithms for training and prediction can be applied based on the selected FVs.
The aim of FVS is to represent the mapping of all the training data points in RKHS with a linear

combination of selected FVs. Suppose $(\boldsymbol{x}_i, y_i)$, for $i = 1, 2, \dots, T$, are the training data points and the mapping $\varphi(\boldsymbol{x})$ maps each input $\boldsymbol{x}_i$ into RKHS with the mapping $\boldsymbol{\varphi}_i$, for $i = 1, 2, \dots, T$; $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \varphi(\boldsymbol{x}_i), \varphi(\boldsymbol{x}_j) \rangle$ is the inner product between $\boldsymbol{\varphi}_i$ and $\boldsymbol{\varphi}_j$.

In order to find a new FV, we just need to verify if the mapping $\boldsymbol{\varphi}_N$ of a new data point $(\boldsymbol{x}_N, y_N)$ can be represented by a linear combination of the existing FVs. Suppose the existing FVs are included in the feature space $\mathbf{S} = \{\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2, \dots, \boldsymbol{\varphi}_L\}$ and the corresponding original data points are $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_L\}$. The verification of the new FV amounts to finding the vector $\boldsymbol{a}_N = \{a_{N,1}, a_{N,2}, \dots, a_{N,L}\}$ which gives the minimum of (3) below:

$$\mu_N = \frac{\left\| \boldsymbol{\varphi}_N - \sum_{i=1}^{L} a_i \boldsymbol{\varphi}_i \right\|}{\|\boldsymbol{\varphi}_N\|} \tag{3}$$

It is difficult to give the mapping function $\varphi(\boldsymbol{x})$ and make the previous calculation in RKHS. On the other hand, the kernel function gives the inner product of two data points in RKHS without having to know the exact mapping function. Thus, the minimum of μ_N can be expressed by an inner product of the kernel functions

$$\min \mu_N = 1 - \frac{K_{S,N}^t K_{S,S}^{-1} K_{S,N}}{k_{N,N}}, \tag{4}$$

where $K_{S,S}$ is the kernel matrix of $\mathbf{S}$ and $K_{S,N} = (k_{i,N}), i = 1, 2, \dots, L$ is the vector of the inner product between $\boldsymbol{\varphi}_N$ and $\mathbf{S}$; $J_{S,N} = \frac{K_{S,N}^t K_{S,S}^{-1} K_{S,N}}{k_{N,N}}$ is called the local fitness of data point $\boldsymbol{x}_N$ with respect to feature space $\mathbf{S}$. If $1 - J_{S,N}$ is smaller than the pre-set positive threshold ρ (the first tolerance parameter here introduced) for local fitness, the new point is not a new FV, otherwise, it is added to $\mathbf{S}$ as a new FV.

The linear independence between all FVs is necessary and sufficient to make sure that $K_{S,S}$ is revertible. There is no need to further check if the $K_{S,S}$ with the newly added FV is invertible as the original work of [1]. Tolerance parameter ρ controls the number of selected FVs and can decrease the influence of the noise in the data. Its best value is dependent on the hyperparameter in the kernel function, e.g. for RBF, the best ρ for a bigger σ is normally smaller. Large values of ρ lead to less FVs, and vice versa. A good choice of the value of ρ can decrease the noise in the model, while keeping enough FVs to guarantee good performance of the SVR model.

From (4), it is clear that the best values $\boldsymbol{a}_N$ are:

$$\boldsymbol{a}_N = K_{S,N}^t K_{S,S}^{-1}. \tag{5}$$

We introduce also the global fitness J_S on the dataset:

$$J_S = \sum_{i=1}^{M} J_{S,i}. \tag{6}$$

Geometrically, FVS is to select the coordinate vectors in RKHS. Figure 1 is an example of a bi-dimensional feature space. Any pair of two linearly independent vector, e.g. $\boldsymbol{\varphi}_1$ and $\boldsymbol{\varphi}_2$ can be

seen as coordinate vectors which form an oblique coordinates system and any other vectors, e.g. $\boldsymbol{\varphi}_3$ can be represented in this space as $a_{31}\boldsymbol{\varphi}_1 + a_{32}\boldsymbol{\varphi}_2$, with $[a_{31}, a_{32}]$ calculated by (5) and $a_{31}\boldsymbol{\varphi}_1$, $a_{32}\boldsymbol{\varphi}_2$ its oblique projections on $\boldsymbol{\varphi}_1$ and $\boldsymbol{\varphi}_2$. For a vector, e.g. $\boldsymbol{\varphi}_4$ outside the bi-dimensional feature space, the closest vector to this vector in the feature space is $\boldsymbol{\varphi}_5$ which is its projection on this space of; then, $a_{41}\boldsymbol{\varphi}_1$, $a_{42}\boldsymbol{\varphi}_2$ are the oblique projections of $\boldsymbol{\varphi}_5$ on $\boldsymbol{\varphi}_1$ and $\boldsymbol{\varphi}_2$, with a_{41}, a_{42} calculated with (5). Thus, for any vector $\boldsymbol{\varphi}$ in RKHS, its local fitness is $\cos^2\theta$, with θ the angle between this vector and the feature space. If $\boldsymbol{\varphi}$ is in this feature space, θ is 0, otherwise θ is in the interval $(0, \pi/2]$. The threshold ρ assures that only the vector whose θ is bigger than $\arcsin\sqrt{1-\rho}$ is selected as the next new feature vector. The function of ρ is like the ε in the ε-insensitive loss function of SVR.

Figure 1

2.3 Incremental and Decremental learning

Incremental & Decremental Learning as proposed in [4], provides a good "tool" for SVR to adaptively modify the SVR model with new data points. The idea is to find the Kuhn-Tucker conditions for a new data point by iteratively modifying its influence in the regression function while keeping the Kuhn-Tucker conditions satisfied by the other data points in the model. This method can "adiabatically" add a new point and remove an existing point in the SVR model, instead of retraining it from the beginning. Although it has been proposed for classification problems in the original work, the method has been applied also for regression problems [16].
In this paper, Incremental & Decremental Learning is used for the tasks of ADDITION (add a new FV) and UPDATE (update the output of an existing FV) in the model, after some necessary verifications.

2.4 Online-SVR-FID

Figure 2

The Online-SVR-FID method can be divided into two parts: one is Offline Training, i.e. selecting FVs in the available offline data and training the SVR model; the other is Online Learning, i.e. for each new data point, verifying if it is a new pattern, a changed pattern or just an existing pattern and taking the corresponding action. Figure 2 shows the paradigm of Online-SVR-FID. The pseudo-code is given in Figure 3.

Figure 3

2.4.1 Offline Training of Online-SVR-FID

Offline Training includes two steps. The first step is to select the FVs in the training dataset with FVS. The aim is to find the feature space **S** formed by part of the training dataset, which gives the minimum of the global fitness J_S calculated with (6) on the whole training dataset $\boldsymbol{T}_r$. As shown in Fig. 3, the procedure is an iterative process of sequential forward selection. For the first iteration,

the data point which gives the minimum of the global fitness $J_{\boldsymbol{S}}$ on $\boldsymbol{T}_r$ is selected as the first FV in the feature space **S**. The following iterations are the same: the next possible FV is the point in the reduced training dataset $\boldsymbol{T}_r$, which gives the maximum of the local fitness with the current feature space $\boldsymbol{S}$; if $1 - J_{\boldsymbol{S},k}$ for this point is bigger than the predefined threshold ρ, the data point is added to **S** as FV and the training dataset is reduced as $\boldsymbol{T}_r = \boldsymbol{T}_r \backslash \mathbf{E}$ with $\mathbf{E} = \{(\boldsymbol{x}_k, y_k)\ and\ (\boldsymbol{x}_i, y_i): 1 - J_{\boldsymbol{S},i} \leq \rho\}$; otherwise, the FV selection in the training dataset is finished. In the FVs selection process, the calculation of the local fitness of each data point in $\boldsymbol{T}_r$ is most time-consuming, and, thus, at the end of each iteration, the training dataset $\boldsymbol{T}_r$ is reduced by deleting the data points that can not be new FVs in the next iteration. The deleted data points are the one which is selected as new FV in the current iteration and those whose local fitness satisfies $1 - J_{\boldsymbol{S},i} \leq \rho$, because the feature space $\boldsymbol{S}$ in the next iteration contains one more FV and, then, their local fitness in the next iteration is smaller or at least equal to their local fitness in this iteration. Compared to searching the next possible FV in the whole training dataset, as proposed in [11], the FV selection process proposed in this paper takes less computation time. The second step is to train a SVR model with FVs in **S** using a classical algorithm. The data points used to form the final function in (2) are only the selected FVs, but the objective function in (1) is still to be minimized on the whole training dataset. Such quadratic optimization setting can in a sense avoid the overfitting problem bothering SVR.

In [2], each time a new data point is selected as FV, it is added to the model only if the matrix $K_{\boldsymbol{S},\boldsymbol{S}}$ in (5) is invertible after adding the new data point into $\boldsymbol{S}$. In fact, this is not necessary: if $1 - J_{\boldsymbol{S},N} > \rho$, the FVs, including the new data points, are linearly independent, which ensure that $K_{\boldsymbol{S},\boldsymbol{S}}$ is invertible; thus, in this paper, $1 - J_{\boldsymbol{S},N} > \rho$ is the only condition for the verification of new FVs during Offline Training and for the addition into the present model during Online Learning.

2.4.2 Online Learning of Online-SVR-FID

Online Learning consists of detecting new or changed patterns considering, respectively, the characteristics of the inputs and the bias of the prediction of the new data points and, then, carrying out the ADDITION and UPDATE tasks, as illustrated in Fig. 3. In general, verification of the linear independence between the mapping of the new input and the existing FVs in the feature space **S** is used to verify if the new point is a new FV (pattern). The difference (bias) between the predicted value and the real output of the new data point is used to decide the change of the existing patterns. Suppose a new data point is $(\boldsymbol{x}_N, y_N)$ and the prediction model for this instance is M trained on feature space **S**. The first step is to verify if $(\boldsymbol{x}_N, y_N)$ is a new pattern by calculating its local fitness $J_{\boldsymbol{S},N}$ with (4), i.e. to verify if the mapping $\boldsymbol{\varphi}_N$ of $(\boldsymbol{x}_N, y_N)$ can be expressed by a linear combination of all FVs in **S**. If $1 - J_{\boldsymbol{S},N}$ is bigger than the predefined threshold ρ, i.e. the linear combination of FVs in **S** cannot sufficiently approximate $\boldsymbol{\varphi}_N$, $(\boldsymbol{x}_N, y_N)$ is taken as a new pattern and added directly to the model using Incremental Learning as in [4]; the model M and the feature space **S** are updated at the same time and await for the next new data point without going to the second step of checking the bias of the predicted values compared to the true output. Otherwise i.e. $1 - J_{\boldsymbol{S},N} \leq \rho$, it is not a new pattern and we proceed to the second step to verify if there is any change in the existing patterns.

The second step of online learning feeds the new data point to the model and calculates the difference between the predicted value using M and the real output y_N of the new data point, i.e. bias $= |\widehat{y_N} - y_N|$, with $\widehat{y_N}$ the predicted value of the new data point. If the bias is smaller than the predefined threshold δ (the second tolerance parameter here introduced), there is no change in

the existing patterns and the model M is kept unchanged and awaits for the next new data point; otherwise, one or several existing patterns in M have changed and it or they need to be updated.

In practice, it is not always easy to identify the changed patterns, as the pattern related to the new data point can be expressed as a linear combination of all the existing patterns and it is hard to find out which is (are) changed. The idea proposed in this paper is using the new data point to replace one specifically selected data point in M. The procedure is as follows:

1. A vector $\boldsymbol{m} = (m_1, m_2, \dots, m_l)$ is used to record the contribution of each FV to the SVR models. Each value in $\boldsymbol{m}$ corresponds to a FV in the model.
2. $\boldsymbol{m}$ is set to be a zero vector before Offline Training.
3. When the model M is trained during Offline Training with the selected FVs from the training dataset, m_i is increased by 1 if the corresponding FV is a SV, i.e. its multiplier in (2) is not zero. Otherwise, i.e. for a FV with zero multiplier, its contribution m_i is zero.
4. Each time the model is added with one new data point, a new m_{l+1} is added to $\boldsymbol{m}$ to record the contribution of the new FV in the model. After the model is updated with ADDITION, the contribution m_i of each FV in the model is updated with the contribution update rules: if the data point is a SV in the new updated model, its new contribution is calculated as $m_i^{new} \leftarrow \tau * m_i + 1$, with τ a positive constant smaller than 1, i.e. the contribution of a FV in the new model is more weighted than that in the old models; otherwise it is kept unchanged.
5. When a change is detected with respect to the old patterns, the first step is to calculate the values $\boldsymbol{a}_N$ for the new data point according to (5). Then, among all the FVs in the model with non-zero values in $\boldsymbol{a}_N$, the one with least contribution, say m_I, is deleted from the model using Decremental Learning as in [4] and m_I is reset to zero. If there are several FVs with the same contribution and the least contribution, the FV to be replaced is selected as the oldest one among them.
6. The new data point is added to the model using Incremental Learning in [4] and it inherits the contribution m_I, which is zero for now. The vector $\boldsymbol{m}$ and the feature space **S** are updated, and also the contribution of the FV is updated according to the rules in step 4 above.

Note that the FV in the model with least contribution to the SVR models among all those with non-zero values in the linear combination (according to (5)) is replaced by the new data point. This strategy for updating a changed pattern must and can keep the FVs in the model linearly independent, so that the Kernel matrix $K_{S,S}$ in (4) is invertible and the Online Learning can continue to be carried out. If a new pattern is added because of the noise, this strategy can decrease the influence of the new data points and keep the capability of the model, as only one existing FV with least contribution is replaced. Note also that if a new data point is a new pattern, it is added instantly in the model, without consideration of the bias of its prediction, so that a maximal richness of the patterns are kept in the model. This is different from the online learning methods which consider only the prediction accuracy. The changed patterns are made of the points which can be expressed as a linear combination of existing patterns in RKHS, but with a bias of prediction larger than the preset threshold δ. This allows replacing a changed pattern instead of adding it in the model, in order to keep the FVs in the model linearly independent and up-to-date.

Note that proper selection of the (positive) values for the tolerance parameters, ρ and δ, can efficiently decrease the influence of noise and avoid overfitting by selecting only informative parts of the dataset.

3. Artificial and real case studies

In this section, Online-SVR-FID is compared with four other online learning methods: original Incremental Learning in [4], Naïve Online Reg Minimization Algorithm (NORMA) in [12], SOGP in [3] and Kernel-based Recursive Least Square Tracker (KRLS-T) in [14]. Details for these online learning approaches can be found in the related literature.
The procedure for adopting Online-SVR-FID is illustrated in Figure 4, including data preprocessing, data reconstruction, tuning hyperparameters, offline training and online learning.

Figure 4

3.1 Experiments on artificial datasets

A comparison is made on four artificial drifting datasets to test the generalizability and robustness of the proposed approach.

3.1.1 Data description

Four artificial datasets are generated for the case study. The Friedman's function is widely used to generate the artificial datasets with pattern drifts [27]. From (5), we can see that there are five input variables and one output variable. Five other input variables are also included in the inputs, which are not related to the output. These five input variables follow a uniform distribution over the interval $[0,1]$. The complete structure of the data point is $(\boldsymbol{x}, y) = \left((x^1, \dots, x^{10}), y\right)$.

$$y_i = 10 * \sin\left(\pi x_i^1 x_i^2\right) + 20 * \left(x_i^3 - 0.5\right)^2 + 10 * x_i^4 + 5 * x_i^5 \quad (5)$$

According to [28], three drifting datasets are produced, including global recurring abrupt drift dataset (GRA) with global, abrupt, and recurring drifts introduced in two drift points, global non-recurring gradual drift (GnRG) dataset with gradual drifts introduced from two data points, local and abrupt drift dataset (LA) with three abrupt drifts. A total of M = 1000 data points are generated for each dataset.
The fourth drifting dataset (Hyperplane) is generated similarly to [29], which includes four different concepts and generates totally M (=1000) data points; the dataset includes nine input variables that are uniformly distributed over the interval $[0,1]$.

Concept 1: $y_i = (x_i^1 + x_i^2 + x_i^3)/3$, for $i = 1, \dots, \frac{M}{4}$;

Concept 2: $y_i = (x_i^2 + x_i^3 + x_i^4)/3$, for $i = \frac{M}{4} + 1, \dots, \frac{2M}{4}$;

Concept 3: $y_i = (x_i^4 + x_i^5 + x_i^6)/3$, for $i = \frac{2M}{4} + 1, \dots, \frac{3M}{4}$;

Concept 4: $y_i = (x_i^7 + x_i^8 + x_i^9)/3$, for $i = \frac{3M}{4}, \dots, M$.

For these four artificial datasets, a noise generated by a Gaussian distribution with zero mean is added to the output and each input variable. One can add the same or different noises to the output and inputs by choosing different variances of the Gaussian distribution.

3.1.2 Results comparison

For these four datasets, the output values are firstly normalized to $[0,1]$, and each output value is disturbed by a noise randomly generated from a zero-mean Gaussian distribution with a variance of 0.1. We denote $\boldsymbol{x}\_real$ the inputs without noise, y_real the output without noise and $y\_0.1$ the noisy output with the noise generated randomly from a zero-mean and 0.1-variance Gaussian distribution. The first 250 data points $(\boldsymbol{x}\_real, y\_0.1)$ form the training dataset and the rest 750 data points $(\boldsymbol{x}\_real, y\_real)$ are the test dataset.

Table I shows the prediction results of Online-SVR-FID and the other four benchmark methods, considering the Mean Squared Error (MSE) and Mean Absolute Relative (MARE) between the predicted values and y_real, and the computation time for the test dataset using a computer with an Intel Core i5-2450M 2.50 GHz process of 2 cores and 3GB of RAM. The bolded values are the best prediction results given by all approaches.

Table I

From Table I, we can see that Online-SVR-FID gives comparable results, with the best results obtained by the benchmark methods for all the datasets. Compared to the original Incremental Learning method proposed in [4], Online-SVR-FID uses much less time, because of the integration of FVS to decrease the training data points.

Further experiments are carried out to compare the robustness of Online-SVR-FID and the best approach (KRLS-T) of the benchmark methods. Different levels of noise are added to the input variables and output. The noise follows a zero-mean Gaussian distribution. The variances of the Gaussian distribution generating the noise in the input and output variables is chosen as [0, 0.02, 0.05, 0.1] and [0, 0.05, 0.1], respectively. The settings of the training dataset and test dataset are the same as the previous experiment.

Tables II and III

Tables II and III show the results (MSE between the prediction and the real output under different noise levels in inputs and output) for datasets GnRG and GRA. The Tables show that under the same noise in the output, the MSEs are nearly the same for different noises in the inputs, while the inverse is not true, i.e. the noise in the output degrades the prediction results more severely than the noise in the inputs.

Figures 5 and 6, respectively, show the boxplot of the MSEs of Online-SVR-FID and KRLS-T on datasets GnRG and GRA listed in Tables II and III. Online-SVR-FID gives more stable results than KRLS-T under different noise levels, although the prediction results are slightly degraded compared to KRLS-T. This is caused by the FVS integrated in Online-SVR-FID, which decreases the number of training data points, thus, increasing the robustness of the model at the sacrifice of accuracy.

In Table III, it is observed that, for Online-SVR-FID, with 0 noise input, the prediction for output with a noise of variance 0.1 is slightly better than that with a noise of variance 0.05. This is not intuitive and is caused by the grid search method for tuning hyperparameters. Few other contradictions may be found for the same reason in Table III, but these do not impair the conclusions drawn from the comparison of the robustness of the two methods.

Figures 5 and 6

3.2 Experiment on a real case study

3.2.1 Data description

In this part, time series data collected by a sensor for measuring the leak flow in the first seal of the Reactor Coolant Pump (RCP) in a NPP are used to test the performance of the proposed Online-SVR-FID approach. The RCP is a fundamental component for the safe operation of a NPP. Its function is to provide cooling water into the reactor, to remove the heat produced by nuclear fission. The leaked radioactive water from the RCP may endangers the personnel in NPP and, most of all, a large amount of leakage reduces the cooling effect, with the risk of material melt down with catastrophic results. Thus, it is critical to monitor and predict the leak flow of RCP.

The specific objective considered in this paper is to predict the future evolution of the leak flow, so as to anticipate when its value will reach certain thresholds of alarm which demand interventions, such as shut-down and maintenance: in short, it is a prognostics problem and the approach taken is that of data-driven modelling for prediction [24].

3.2.2 Tuning hyperparameters

The ε-insensitive loss function and RBF kernel function are used to build the SVR model for the prediction. There are five unknown parameters to be set: three hyperparameters in SVR σ, ε, C and two tolerance parameters ρ, δ. the parameter σ is calculated with (7) as proposed in Cherkassky and Ma (2004); the parameter μ is a value between 0 and 1; the parameter δ=0.05 is given by the expert in EDF according to the operation manual:

$$\sigma^2 = \mu * max\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2, i, j = 1, \dots, T. \qquad (7)$$

With the determined σ and δ, the values of ε and C are set using a grid search method proposed in [15], which minimizes the MSE on the whole training dataset instead of only on the selected FVs. By applying the hierarchical, nonparametric sequential change-detection test proposed in [26], changes are detected at the time steps 420 and 780, which confirms that pattern drifts exist in this real time series data. Partial autocorrelation analysis can, then, help to reconstruct the time series data. After the reconstruction of the raw data (ten historical target values chosen by way of a partial autocorrelation analysis are used as inputs and the target value one day ahead is the output [15], 300 data points are selected as original offline training dataset and the following 500 data points form the test data, which are fed to the model one by one emulating the online learning process. The outputs of the training and testing datasets are shown in Fig. 7 with the first 300 values of the training dataset and the last 500 values belonging to the test dataset. It is clear that the training dataset represents the normal (stable) process, while the testing dataset is the abnormal (increasing) process. This experiment is to verify how fast and accurate Online-SVR-FID can follow the changing trend in the time series data. The experimental results of the proposed online learning approach are here presented. Comparisons with other online learning approaches for kernel-based regression methods proposed in [4], [12], [3] and [14] are carried out and presented in the next Section.

In supervised learning, the performance of SVR is highly dependent on the size of the training dataset. In Online-SVR-FID, the number of FVs in the training dataset is selected by FVS, where

parameters σ (or μ) and ρ are critical, as shown in the pseudo-code in Fig. 3 and Fig. 8. Figure 9 shows the change of MSE on the whole test dataset with different values for μ and ρ in Online-SVR-FID. For the same value of μ, smaller values of ρ select more training data points as FVs, which leads generally to more accurate prediction results. From Fig. 9, we can also see that when the value of μ is small (i.e. small σ), e.g. $\mu = 0.001$, different values of ρ give very different prediction performances, as the number of selected FVs can be only 1 for bigger values of ρ. But when μ is big enough, e.g. $\mu = 1.3$, different values of ρ give similar prediction results, better than for smaller μ: thus, the value of μ is critical.

Note that in this real case study of online learning, it can be seen that more FVs selected from the training dataset with bigger μ and smaller ρ do not always improve the prediction significantly: in this case study, when the number of FVs is larger than 10, the prediction results are comparable. This proves that the dimensionality of the training dataset in RKHS is fixed and the few selected FVs can represent the whole training dataset.

Figures 7, 8, 9, 10

3.2.3 Prediction results of Online-SVR-FID

The time for Online Learning of the test dataset is dependent on the number of FVs. The more FVs are selected from the training dataset, the more time is needed for training a SVR model and Online Learning. Thus considering the prediction accuracy and the computational burden, the best values for μ and ρ are taken as 10^{-3} and $2.2*10^{-8}$ for the case study. The MSE and computational time are 0.0011 and 8.8944s. The values of ε and C are 0.0152 and $1.5199*10^{4}$. A total of 11 data points from the original training dataset are selected as FVs and used to train a SVR model. The prediction results on the test dataset using Online-SVR-FID are shown in Fig. 10 with the positions of new patterns (ADDITION, marked by $\diamond$ in the Figure) and changed patterns (UPDATE, marked by $\square$ in the Figure) indicated by symbols. The online-SVR-FID treats the data points from the test dataset one by one, simulating the online learning procedure.

After the online learning process with Online-SVR-FID, 3 and 53 data points in the test dataset are selected respectively for ADDITION and UPDATE. Note that only ADDITION changes the size of the model, so the number of data points in the final model is 14, which is far smaller than the total number of training and test data points, which is 800.

Note that based on the previous analysis of the impact of μ and ρ, the tuning of ρ can be simplified, since for a fixed value of μ we can calculate the change of MSE on the training dataset for decreasing values of ρ; the best value for ρ is the one for which the decrease of MSE is smaller than a given threshold.

3.2.4 Results Comparison

The offline SVR model with RBF kernel function and ε-insensitive loss function trained on the 300 data points of the training dataset using the method proposed in [15] serves as the initial model before online learning for Incremental Learning and NORMA. The values for hyperparameters (C, ε, σ) in SVR are (10000, 0.0025, 0.100). The learning rate η in NORMA is set to be $5*10^{-6}$, as $C\eta$ in (8) should be smaller than 1. Truncation is proposed in [12] to control the size of the model and the truncation threshold is 0.01, i.e. the training data points with multipliers in (2) smaller than 0.01 are deleted from the model.

A model is trained on the 300 training data points by SOGP and, then, each time a new data point is

available, it is added to the training dataset and the model is updated as proposed in [3]. In SOGP, the threshold for new basis vector is 10^{-8}, and σ in RBF is 0.01 while the maximal number of basis vectors is 100.

In the algorithm of KRLS, the width of the RBF kernel function is set to be 0.1. The forgetting rate is 0.999, and the budget (maximal number of data points in the model) is fixed at 200.

The comparisons of prediction results (MSE, MARE) and computation complexity (time for online learning, model size before Online Learning, model size after Online Learning) using the same computer (Interl Core i5 @ 2.5 GHz CPU and 4G RAM) are reported in Table IV.

Table IV

3.2.4.1 Computational Complexity

As a way to evaluate the computational complexity, we compare the times of Online Learning. The time for Offline Training is not considered, because there are different methods for parameter tuning for the different approaches, which influence the Offline Training time. What is more, since we consider Offline Training & Online Learning with a focus on the latter, the time for Offline Training is not critical for Online Learning: the relevant part in the present work is that the approach can learn the new patterns efficiently during Online Learning.

In the real case study considered, the proposed Online-SVR-FID is seen (Table IV) to use significantly less time, due to a much reduced model size, while achieving comparable accuracy in the prediction.

Indeed, the computation time of these four methods during Online Learning depends highly on the model size: thus, reducing the number of data points in the model means reducing the computational complexity during online learning. In Online-SVR-FID, the ADDITION process for new patterns increases the model size whereas the UPDATE process for changed patterns just changes data points in the model while keeping the model size (number of FVs) unchanged. Such an Online Learning mechanism makes the size of model much smaller than those of the other four benchmark methods. NORMA uses only the recent data points (a maximal number of 269), like a sliding time window approach. Incremental Learning adds each new data point in the model. SOGP adds all data points in the model and, then, uses a sparseness strategy to delete some randomly selected data points, which can be expressed as a linear combination of the rest of the data points in RKHS; thus, it decreases greatly the size of the model, but still consumes much more time than Online-SVR-FID, as this latter modifies the model only with previously selected data points. Although the number of data points in the KRLS-T model before and after Online Learning is both 200, which is bounded by the budget and is much larger than those of Online-SVR-FID and SOGP, the time used for Online Learning of 500 data points is much less than SOGP and comparable with Online-SVR-FID. The experiment shows that KRLS-T uses the same time as Online-SVR-FID, while its model size is uch larger than Online-SVR-FID. This is because the Incremental & Decremental Learning in Online-SVR-FID is an iterative process to update the multipliers in kernel expansion while the adaptation of a KRLS-T is directly (and rapidly) calculated analytically.

One advantage of SOGP, NORMA and KRLS-T is that they can give an upper bound of the size of the model in the case of infinite new data points, while Online-SVR-FID is not able to give such a bound. But the following theorem states that the number of FVs for Online-SVR-FID is finite which is similar to [19].

Theorem 1 Let $k: X \times X \to R$ be a countinous Mercer kernel, with X a compact subset of a Branch space. Then, for any training sequence $\Gamma = \{(\boldsymbol{x}_i, y_i)\}, i = 1,2, \dots, T$ and for any tolerance parameter $\rho > 0$, the size of the FVs of Online-SVR-FID is finite, even if the number of new data points grows to infinite with time.

Proof The proof of this theorem can be easily derived with from proof of Theorem 3.1 in [8] and Theorem 1 in [18]. With the Mercer theorem, there exists a mapping $\boldsymbol{\varphi}: X \to \mathrm{H}$, where H is a RKHS. $k(\boldsymbol{x}, \boldsymbol{x}^*)$ and $\boldsymbol{\varphi}(\boldsymbol{x})$ is continuous. Given that X is compact, it is natural that $\boldsymbol{\varphi}(X)$ is compact too. Each time a new FV $(\boldsymbol{x}, y)$ is added to the feature space S with L FVs, we have

$$\rho^2 \leq min_{\boldsymbol{a}} \frac{\left\|\boldsymbol{\varphi}_N - \sum_{i=1}^{L} a_i \boldsymbol{\varphi}_i\right\|^2}{\|\boldsymbol{\varphi}_N\|^2} \leq \frac{\|\boldsymbol{\varphi}_N - \boldsymbol{\varphi}_i\|^2}{\|\boldsymbol{\varphi}_N\|^2},$$

for any $i = 1,2, \dots, L$. The definition of packing numbers in [25] shows that the maximum number of FVs in Online-SVR-FID is bounded by the packing number at scale ρ of $\boldsymbol{\varphi}(X)$, while this number is smaller than the covering number at scale $\rho/2$ which is finite with a compact set.

3.2.4.2 Prediction accuracy

With respect to the prediction accuracy, NORMA gives the worst results in the case study considered. The performance of NORMA decreases with the online learning process. The update strategy of NORMA for the multipliers in (2) destroys the properties of SVR, i.e. the multipliers do not satisfy the Kuhn-Tucker conditions after the update procedure. The multipliers for the new data points are set to be the positive or negative values of the learning rate, which can be too small compared to the non-zero multipliers derived by the Kuhn-Tucker conditions, which are comparable to the penalty factor C in SVR, as the optimal value of C is very large in this case study. Such setting makes the contribution of the new data points negligible compared to the other data points in the model. Thus, the model does not catch effectively the new patterns and cannot perform well on the new data points, nor on the previous data points.

With the fastest Online Learning speed, KRLS-T gives slightly worse results than Online-SVR-FID, Incremental Learning and SOGP, while the latter three methods are giving comparable results. The post-processing for sparseness in SOGP is carried out in a random way, i.e. a randomly selected data point is deleted if it can be expressed by a linear combination of the rest; otherwise, it is kept. This randomness leads to unstable prediction results for SOGP in this case study. For example, in the case of changed patterns, any of them can be expressed as a linear combination of the rest; if the sparseness process deletes the ones more informative to the future patterns, the model can no longer perform well on the selected pattern.

In conclusion, on the real case study, the proposed Online-SVR-FID significantly reduces the online learning time and can learn timely and efficiently the new and changed patterns; it gives comparable or even better results than the benchmarks considered.

4. Conclusions

In this paper, we have proposed an online learning approach for SVR, named Online-SVR-FID, to

efficiently address the pattern drift problem by online learning.

Four artificial datasets and a case study concerning one real time series data of leakage from the first seal of RCP in a NPP have been considered. The application of the proposed approach shows that it is capable of reducing the number of data points in the model, and timely learning the incoming patterns by ADDITION (new patterns) and UPDATE (changed patterns), when necessary. Two tolerance parameters ρ and δ are introduced to reduce the influence of the noise and to control the number of actions of ADDITION and UPDATE in the learning process. Compared in terms of MSE and MARE on the test datasets with other benchmarks for online learning i.e. NORMA, SOGA, KRLS-T and Incremental Learning, Online-SVR-FID has been shown to be effective on the case studies considered, with accuracy comparable to that of the best benchmark method. Furthermore, the prediction results for GnRG and GRA artificial datasets under different noises in the inputs and output show that the proposed approach is more robust to the noise than KRLS-T.

While it is true that a number of papers have already presented solutions for the reduction of the training dataset by forward or backward selection of a smaller number of feature (or representative) vectors, in this paper the main novelty lies in the proposed update strategy based on the special method of FVS under a nonstationary environment. The proposed update strategy considers both the geometrical relations between different data points in the Reproduced Kernel Hilbert Space (RKHS) and the prediction accuracy. Special strategies are proposed for two different kinds of patterns drifts (as defined in the Introduction of the paper, new patterns and changed patterns). In the experiments, the proposed approach gives more robust results than KRLS-T. Compared to SOGP, the online approach proposed in this paper cannot bound the data points in the model, but the Theorem 1 introduced in the paper proves that the number is finite in the case of infinite data points.

Future work will be devoted to further testing on other real datasets that will become available from the application field.

References

[1] G. Baudat, F. Anouar, Feature vector selection and projection using kernels, Neurocomputing. 55 (2003) 21–38. doi:10.1016/S0925-2312(03)00429-6.

[2] A. Bordes, Ş. Ertekin, J. Weston, L. Bottou, Fast Kernel Classifiers with Online and Active Learning, J. Mach. Learn. Res. 6 (2005) 1579–1619. doi:10.1.1.60.9676.

[3] L. Csató, M. Opper, Sparse on-line gaussian processes., Neural Comput. 14 (2002) 641–668. doi:10.1162/089976602317250933.

[4] G. Cauwenberghs, T. Poggio, Incremental and Decremental Support Vector Machine Learning, in: NIPS, 2000: pp. 409–415. http://citeseer.ist.psu.edu/cauwenberghs00incremental.html.

[5] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, Y. Singer, Online Passive-Aggressive Algorithms, J. Mach. Learn. Res. 7 (2006) 551–585. doi:10.1.1.9.3429.

[6] S.-L. Dai, C. Wang, M. Wang, Dynamic learning from adaptive neural network control of a class of nonaffine nonlinear systems., IEEE Trans. Neural Networks Learn. Syst. 25 (2014) 111–23. doi:10.1109/TNNLS.2013.2257843.

[7] Q. Liu, M. Dong, W. Lv, X. Geng, Y. Li, A novel method using adaptive hidden semi-Markov model for multi-sensor monitoring equipment health prognosis, Mech. Syst. Signal Process. 64-65

(2015) 217–232. doi:10.1016/j.ymssp.2015.03.029.
[8] Y. Engel, S. Mannor, R. Meir, The kernel recursive least-squares algorithm, Signal Process. IEEE Trans. 52 (2004) 2275–2285. doi:10.1109/TSP.2004.830985.
[9] J.B. Gao, S.R. Gunn, C.J. Harris, M. Brown, A probabilistic framework for SVM regression and error bar estimation, Mach. Learn. 46 (2002) 71–89. doi:10.1023/A:1012494009640.
[10] T. Jung, D. Polani, Sequential Learning with LS-SVM for Large-Scale Data Sets, Lect. Notes Comput. Sci. (2006) 381–390.
[11] M. Karasuyama, I. Takeuchi, Multiple incremental decremental learning of support vector machines, IEEE Trans. Neural Networks. 21 (2010) 1048–1059. doi:10.1109/TNN.2010.2048039.
[12] J. Kivinen, A.J. Smola, R.C. Williamson, Online learning with kernels, IEEE Trans. Signal Process. 52 (2004) 2165–2176. doi:10.1109/TSP.2004.830991.
[13] L.I. Kuncheva, W.J. Faithfull, PCA feature extraction for change detection in multidimensional unlabeled data, IEEE Trans. Neural Networks Learn. Syst. 25 (2014) 69–80. doi:10.1109/TNNLS.2013.2248094.
[14] S. Van Vaerenbergh, M. Lazaro-Gredilla, I. Santamaria, Kernel recursive least-squares tracker for time-varying regression, IEEE Trans. Neural Networks Learn. Syst. 23 (2012) 1313–1326. doi:10.1109/TNNLS.2012.2200500.
[15] J. Liu, R. Seraoui, V. Vitelli, E. Zio, Nuclear power plant components condition monitoring by probabilistic support vector machine, Ann. Nucl. Energy. 56 (2013) 23–33. doi:10.1016/j.anucene.2013.01.005.
[16] J. Ma, J. Theiler, S. Perkins, Accurate on-line support vector regression, Neural Comput. 15 (2003) 2683–2703. doi:10.1162/089976603322385117.
[17] M. Martin, On-line support vector machine regression, Machine Learning: ECML Springer Berlin Heidelberg, 2002(2002) 282-294.
[18] J. Ni, C. Zhang and S. X. Yang, An Adaptive Approach Based on KPCA and SVM for Real-Time Fault Diagnosis of HVCBs, IEEE Trans. Energy Delivery, 26 (2011), 1960-1971.
[19] F. Orabona, J. Keshet, B. Caputo, Bounded Kernel-Based Online Learning, J. Mach. Learn. Res. 10 (2009) 2643–2666.
[20] R. Elwell, R. Polikar, Incremental Learning of Concept Drift in Nonstationary Environments, IEEE Trans. Neural Networks. 22 (2011) 1517–1531. doi:10.1109/TNN.2011.2160459.
[21] D. Petković, S. Shamshirband, A. Abbasi, K. Kiani, E.T. Al-Shammari, Prediction of contact forces of underactuated finger by adaptive neuro fuzzy approach, Mech. Syst. Signal Process. 64-65 (2015) 520-527. doi:10.1016/j.ymssp.2015.03.013.
[22] I. Khelf, L. Laouar, A.M. Bouchelaghem, D. Rémond, S. Saad, Adaptive fault diagnosis in rotating machines using indicators selection, Mech. Syst. Signal Process. 40 (2013) 452–468. doi:10.1016/j.ymssp.2013.05.025.
[23] D. Wang, B. Zhang, P. Zhang, H. Qiao, An online core vector machine with adaptive MEB adjustment, Pattern Recognit. 43 (2010) 3468–3482. doi:Doi 10.1016/J.Patcog.2010.05.020.
[24] A. K.S. Jardine, D. Lin, D. Banjevic, A review on machinery diagnostics and prognostics implementing condition-based maintenance, Mech. Syst. Signal Process. 20 (2006) 1483–1510. doi:10.1016/j.ymssp.2005.09.012.
[25] F. Cucker, D. X. Zhou, Learning Theory: "An Approximation Theory Viewpoint," (2007) Cambridge University Press, New York, NY, USA.
[26] C. Alippi, G. Boracchi, M. Roveri, A hierarchical, nonparametric, sequential change-detection

test, in: Proc. Int. Jt. Conf. Neural Networks, 2011: pp. 2889–2896. doi:10.1109/IJCNN.2011.6033600.

[27] J.H. Friedman, Multivariate Adaptive Regression Splines, Ann. Stat. 19 (1991) 1–67. doi:10.1214/aos/1176347963.

[28] E. Ikonomovska, J. Gama, S. Džeroski, Online tree-based ensembles and option trees for regression on evolving data streams, Neurocomputing. 150 (2015) 458–470. doi:10.1016/j.neucom.2014.04.076.

[29] S. Gomes Soares, R. Araújo, An on-line weighted ensemble of regressor models to handle concept drifts, Eng. Appl. Artif. Intell. 37 (2015) 392–406. doi:10.1016/j.engappai.2014.10.003.