A SVR-based ensemble approach for drifting data streams with recurring patterns

Jie Liu, Enrico Zio

A SVR-based ensemble approach for drifting data streams with recurring patterns

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

[1] Chair on System Science and the Energetic Challenge, Fondation Électricité de France (EDF), CentraleSupélec, Université Paris-Saclay, Grande Voie des Vignes, 92290 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

Pattern drift is a common issue for machine learning in real applications, as the distribution generating the data may change under nonstationary environmental/operational conditions. In our previous work, a strategy based on Feature Vector Selection (FVS) has been proposed for enabling a Support Vector Regression (SVR) model to adaptively update with streaming data, but the proposed strategy suffers from the incapability of treating recurring patterns. An instance-based online learning approach is proposed in this paper, which can adaptively update an SVR-based ensemble model with steaming data points. The proposed approach reduces the computational complexity of the updating process by selecting only part of the newly available data and allows following timely the ongoing patterns by resorting to FVS. The proposed approach creates new sub-models directly from a basic model and the sub-models represent separately the data stream at different periods. A dynamic ensemble selection strategy is integrated in the approach to select the sub-models most relevant to the new data point for deriving the prediction, while reducing the influence of the irrelevant ones. The weights of the different models in the ensemble are updated, based on their prediction errors. Comparison results with several benchmark approaches on several synthetic datasets and on the dataset concerning the leakage from the first seal in a Reactor Coolant Pump, prove the efficiency and accuracy of the proposed online learning ensemble approach.

Key words:
Online learning; ensemble; feature selection; pattern drifts; Support Vector Regression (SVR); Feature Vector Selection (FVS); Reactor Coolant Pump (RCP); seal leakage; Nuclear Power Plant (NPP)

1. Introduction

Predicting malfunctions and failures is a fundamental task for the reliable and safe operation of components and systems. Building an efficient and accurate predictor from available data is one of the main objectives in machine learning. This is particularly critical in high-risk industries, like nuclear, chemical process, aerospace and aviation ones. Regarding the nuclear industry, from which

the case study of this paper is taken, most of the current approaches for prediction of parameters in NPP components and systems, are developed for static environments only [1], [2], [3], i.e. the data are assumed to originate from a process whose underlying distribution does not change with time. On the contrary, in practical applications, the underlying, normally unknown distribution generating the data can vary with time, causing pattern drifts. Pattern drifts in NPP components and systems can be due to a natural evolution of the environment, changes in the operational conditions or faults [4]. In such cases, the data originate from nonstationary environments and models are developed and trained for static environments can no longer give accurate predictions for the new data.

Pattern drifts can be divided into sudden pattern drifts, gradual pattern drifts and recurring patterns. Different approaches have been developed for tackling the different pattern drifts problems, which can be categorized into adaptive single model [5], [6], [7] and online learning ensembles [8], [9], [10]. The former approach is based on an adaptive model that learns incrementally the new patterns and/or forgets the old inefficient ones; however in practice, the computational burden for incremental learning is unacceptable for large datasets, and the recurring patterns are not efficiently handled if they have already been deleted from the model. The online learning ensemble approach aims at updating an ensemble by adapting the sub-models weights and/or adding/deleting a sub-model in the ensemble. The work reported in this paper focuses on this latter approach of online learning ensembles.

There are different types of approaches for online learning ensembles, e.g. data-chunk-based approaches [14], [15], [16], drift detector-based approaches, [18], [19], [20], [21] instance-based approaches, [22], [23], [24] etc.

Accuracy Weighted Ensemble (AWE) [11], Streaming Ensemble Algorithm (SEA) [12], Learning++.NSE [13] are some of the data-chunk-based approaches proposed in the literature. Learning++.NSE trains a new sub-model on the new data chunk if the prediction error exceeds a predefined threshold, and combines the sub-models built through a dynamically modified weighted majority voting. The sub-models weights are calculated based on their weighted-sum performance on different data chunks and added to the ensemble. The problem with these data chunk-based approaches is the determination of the size of the data chunks, as bigger chunks give more stable sub-models but different drifts may be contained in a single sub-model, whereas smaller chunks can better separate different drifts but lead to worse sub-models. There is also a delay in the ensemble for following the ongoing patterns, as the ensemble is updated only when a new data chunk is available and the patterns in the ensemble at this time may no longer be the ongoing patterns.

In order to overcome these difficulties, various approaches have been proposed in the literature, which may combine a drift detector with online learning ensemble to alarm the need for a new sub-model. Adaptive Classifier Ensemble (ACE) [17] slowly builds a new sub-model when the sub-models error on the new data reaches a certain threshold. In [18], pattern drifts are detected by measuring the normalized weighted average output of the sub-models in the ensemble. Diversity analysis is used in [19] to divide different drifts. The most popular drift detector algorithm is the Drift Detection Method (DDM) [20], which models the prediction error on each data point according to a binominal distribution. A new approach for online learning ensembles, called Diversity for Dealing with Drift (DDD) is proposed in [10], which maintains ensembles with different diversity levels. The experimental results show that DDD gives robust and accurate results. Although the drift detector-based approaches can solve the difficulty in deciding a good size of the data chunk, they, compared to instance-based updating approaches, still cannot update the ensemble once a pattern

drift occurs, i.e. sufficient new data are needed before detecting and reacting to the pattern drifts.

In [22], a theoretically supported framework for active learning of drifts in data streams is presented and three active learning strategies are developed based on separate uncertainty, dynamic allocation of labeling efforts over time and randomization of search space. Another instance-based approach, named Online Weighted Ensemble (OWE) is proposed in [25] to learn new data points incrementally in the presence of different types of pattern drifts and to retain old information in recurring patterns. The instance-based updating approaches can learn the pattern drifts effectively and efficiently once they occur. But one main disadvantage is the computational complexity of updating the ensemble with every new data point. Furthermore, a dynamically weighted ensemble is proposed in [26] to store only the features most relevant to the learnt concept, which in turn increases the memory efficiency.

It is important to stress that online learning requires timely updating the ensemble, including its weights and sub-models, while minimizing to the extent feasible the computational burden brought by the updating operations needed for dealing with the different types of drifts.

In this paper, an instance-based online learning approach for Support Vector Regression (SVR)-based ensembles, named Online Ensemble based on Feature Vectors (noted OE-FV, for short), is proposed based on the Feature Vector Selection (FVS) approach presented in [27]. For convenience, in this paper, the term "input vector" refers to the vector containing all the input variables and "input" represents one input variable in the input vector. The term "pattern" refers to one input vector – output pair.

SVR-based ensembles are made of sub-models trained with SVR. FVS calculates the geometrical linear relation among different input vectors of the data points in the Reproduced Kernel Hilbert Space (RKHS) and selects a small part of them as Feature Vectors (FVs) representing the whole training dataset. In our previous work [28], an adaptive online learning approach, named Online-SVR-FID has been proposed for a single SVR model to effectively follow the ongoing patterns by adjusting to two types of drifts (new pattern if the new input vector cannot be represented by a linear combination of the FVs in the model and changed pattern if the new input vector can be represented by a linear combination of the selected FVs but the prediction error is larger than a predefined threshold) and taking the correspondent action. If a new data point is judged as new pattern, it is added directly into the model, while if it is judged as a changed pattern, it is used to replace a selected pattern that makes least contribution to the recent updated models. Compared to several benchmark approaches, Online-SVR-FID has been shown to give comparable results while using much less time. One drawback of Online-SVR-FID is that the old patterns are deleted from the model and one needs to relearn the recurring patterns from scratch.

Based on this previous work, an online learning ensemble is grown from the first kernel-based sub-model $M1$ to store all the past patterns detected in the data, and each sub-model covers patterns in a certain period of the data stream. The ensemble is created sequentially by applying an online learning approach similar to Online-SVR-FID on $M1$. The online learning approach assures that the first sub-model $M1$ follows always the ongoing patterns. If the sub-models $M1$ for different periods of the data stream are separately saved and used in an ensemble, each of them is like an adjusted "copy" of $M1$ tailored to different instances of the online learning process. Every new sub-model is saved by copying the current $M1$ at the time when an old pattern risks of being deleted from the ensemble, as the process for updating $M1$ may use new data points to replace an existing pattern in $M1$ when the new data point is judged as a changed pattern. If the pattern to be

replaced is unique in the ensemble and there is no more such pattern once deleted, the model before the replacement is copied and stored as a new sub-model to guarantee that all the occurred patterns appear at least once in the sub-models of the ensemble, and, then, the updated $M1$ is still the up-to-date sub-model that continues to be updated with future new data points. Note that the sub-models are created sequentially and automatically from $M1$ and are not updated with the new data points. Through the FVS, only data points that are judged as new and changed patterns are used to update $M1$ and create new sub-models when the criterion is reached. Note that each time $M1$ is updated with a new or changed pattern, it is retrained to minimize the MSE on the recent data points. Thus, the computational burden bothering the instance-based approaches for online learning ensemble is reduced. The sub-models weights are updated with each new data point according to the weighted sum of the prediction errors on all the data points, where the prediction errors on the new data points are more weighted than the old ones. Thus, the ensemble can follow efficiently the ongoing patterns. Inspired by the work in [29], [30] and [31], a dynamic ensemble selection strategy is also integrated in the proposed OE-FV. For each new data point, only the most relevant sub-models are used to form an ensemble and derive the weighted-sum prediction result, in order to avoid the influence of the poor ones. The dynamic selection of the sub-models are based on the geometric relation between the input vector of the new data point and the data points in each sub-model. Only the sub-models which can well represent the new input vector are selected.

In order to test the efficiency and accuracy of OE-FV, experiments on several synthetic datasets and a real case study concerning the condition of a component of a Nuclear Power Plant (NPP) are carried out. Comparisons with Learn++.NSE and OWE show that the proposed approach gives better results in the experiments than those of OWE and Learn++.NSE, and the computation time of OE-FV is shorter than that of OWE.

The rest of the paper is structured as follows. FVS and Online-SVR-FID are briefly reviewed in Section 2. Section 3 explains the approach proposed to build the ensemble automatically and the process of weights updating. The experiments on several synthetic datasets and a real case study regarding a component of a NPP are illustrated in Section 4. Comparisons with Learn++.NSE and OWE are also reported in this section. Some conclusions are drawn in Section 5.

2. Brief introduction of FVS and Online-SVR-FID

The proposed online learning approach for kernel-based ensemble is based on the work in [27] and [28]. In order to thoroughly explain the process of building and updating an ensemble with OE-FV, FVS [27] and Online-SVR-FID [28] are firstly and briefly reviewed in this section.

2.1 Feature Vector Selection

Suppose $\boldsymbol{T} = \{(\boldsymbol{x}_i, y_i): i = 1, 2, \dots, M\}$ is the dataset at hand. FVS analyzes the geometric relation among the input vectors of different data points in a high-dimensional space, i.e. RKHS, and selects as FVs the ones which represent the dimensions of the RKHS related to the dataset, in order to decrease its complexity. The other input vectors in the dataset can be represented by a linear combination of the selected FVs in the RKHS. A model can be trained on the selected FVs with classical machine learning methods, e.g. SVR.

In this paper, $\boldsymbol{\varphi}(\boldsymbol{x})$ is the mapping that maps an input vector from the original space to the RKHS

and $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the kernel function that represents the inner product $\langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \boldsymbol{\varphi}(\boldsymbol{x}_j) \rangle$ in the RKHS. Once a new data point $(\boldsymbol{x_n}, y_n)$ with mapping $\boldsymbol{\varphi_n}$ is available, we need to judge if this new data point is a new FV. Suppose the existing FVs selected form the dataset are $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_L\}$ and their mapping are included in the feature space $\mathbf{S} = \{\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2, \dots, \boldsymbol{\varphi}_L\}$; the verification of a new FV amounts to finding the vector $\boldsymbol{a} = \{a_1, a_2, \dots, a_L\}$, which gives the minimum of Equation (1) below:

$$\mu_n = \frac{\|\boldsymbol{\varphi_n} - \sum_{i=1}^{L} a_i \boldsymbol{\varphi_i}\|}{\|\boldsymbol{\varphi_n}\|}. \tag{1}$$

Normally in kernel methods, it is difficult to know the exact expression of the mapping function $\boldsymbol{\varphi}(\boldsymbol{x})$. But kernel functions which represent the inner product between two mappings in RKHS can give a solution to the minimum of Equation (1). The minimum of μ_n can be written as:

$$\min \mu_n = 1 - \frac{K_{S,n}^t K_{S,S}^{-1} K_{S,n}}{k(\boldsymbol{x_n}, \boldsymbol{x_n})}, \tag{2}$$

where $K_{\boldsymbol{S},\boldsymbol{S}}$ is the kernel matrix (gram matrix) of $\mathbf{S}$ and $K_{\boldsymbol{S},\boldsymbol{n}} = (k(\boldsymbol{x}_i, \boldsymbol{x_n})), i = 1,2, \dots, L$ is the vector of the inner product between $\boldsymbol{\varphi_n}$ and $\mathbf{S}$. The derivation of Equation (2) from Equation (1) can be found in [27].

The value calculated with Equation (3) is called local fitness; the definition of global fitness is given in Equation (4); the vector $\boldsymbol{a}$ can be calculated by Equation (5):

$$J_S(\boldsymbol{x_n}) = \frac{K_{S,n}^t K_{S,S}^{-1} K_{S,n}}{k(\boldsymbol{x_n}, \boldsymbol{x_n})} \tag{3}$$

$$J_{\boldsymbol{S}} = \sum_{i=1}^{M} J_{\boldsymbol{S}}(\boldsymbol{x}_i) \tag{4}$$

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

According to the definition of local fitness, each data point is called a pattern and the pattern drifts in this paper are divided into two types: new pattern if the new data point cannot be well represented by the existing FVs in any sub-model, i.e. $1 - J_S(\boldsymbol{x_n}) > \rho$, with ρ a small positive value; changed pattern if the new data point can be represented by the existing FVs in some sub-models, but the predicted value given by all the sub-models is not accurate enough, i.e. $1 - J_S(\boldsymbol{x_n}) < \rho \ \& \ |\hat{y}_n - y_n| > \theta$ for all sub-models, with $\hat{y}_n$ being the predicted value given by one current sub-model and θ a positive value representing the tolerance on the prediction error.

2.2 Online-SVR-FID

Online-SVR-FID proposed in [28] aims at providing an efficient online learning approach for a single SVR model, based on FVS. This approach can be divided into two parts: offline training and online learning.

The offline training is aimed at selecting FVs from the training dataset and training a model on the selected FVs, with the objective of minimizing the Mean Squared Error (MSE) on the whole training dataset.

In [28], it is showed that the number of selected FVs is very critical, as too few FVs cause low accuracy on the test dataset, while too many FVs cause overfitting the training dataset.

The online learning is aimed at detecting the pattern drifts and taking corresponding reactions to update the model. If a new data point is a new FV, it is added to the model and the model is updated; if it is not a new FV and its prediction error is larger than the threshold θ, it replaces the FV which makes least contribution to the recent models.

The case studies in [28] show that Online-SVR-FID gives comparable or even better results than

the SVR trained on the whole training dataset, and Online-SVR-FID uses much less time for online learning the same test dataset. Comparisons with other popular online learning methods are also carried out with respect to the case studies and Online-SVR-FID gives always comparable accuracy. Although the feature selection process increases the computational complexity for data preprocessing, the time for tuning hyperparameters in Online-SVR-FID is much less than the classic SVR model using the whole training dataset, as the Online-SVR-FID model is much less complicated.

Constrained by the length of the paper, the pseudo-code of Online-SVR-FID and the calculation of the contribution of each FV to the recent models are shown in Appendix. Note that Online-SVR-FID can make the model efficiently follow the ongoing patterns in the data. A main drawback of Online-SVR-FID is that some useful FVs are replaced during the UPDATE process shown in Appendix. Once these replaced patterns recur in the new data, the model needs to relearn them, i.e. the information in the past data are not fully stored in the single model.

To overcome this problem, in this paper an ensemble approach is proposed to store all the past patterns in the data and make a reasonable choice of sub-models when facing recurring patterns.

3. The proposed approach for online learning ensemble: OE-FV

As mentioned in previous sections, the main objective of this ensemble is to store all the past patterns in the data, propose strategies to build automatically new sub-models, update their weights and decrease the computational burden for instance-based approaches for online learning ensemble. The whole idea is based on the FVS in [27] and Online-SVR-FID in [28]. OE-FV builds automatically an ensemble from the first sub-model trained on the training dataset. All the sub-models are expected to represent the characteristics of the data during a certain period and once the old patterns recur, the most relevant sub-models are selected by FVS to derive the prediction. As FVS is developed for the kernel methods, OE-FV can be applied for all kernel-based ensembles, e.g. sub-models trained with kernel ridge regression [33], SVR [34], Gaussian process [35] etc.

The main procedure is shown in Figures 1 and 2. OE-FV builds an ensemble sequentially from the first sub-model, named M_1, that is trained on the preliminary training dataset. All the other sub-models can be seen as an updated "copy" of M_1 at one instance during the data stream developing process. These sub-models are expected to be different from each other and represent the data at a certain period of the stream. Only the sub-model M_1 is adaptively updated with new data points, while the other sub-models are fixed once created and stored.

1. Train the first sub-model M_1 with kernel methods on the training dataset.
2. Suppose there are *n* sub-models (M_1, M_2, …, M_n) in the ensemble when a new data point is coming:
   - 2.1 Calculate the predicted value for the new data point by a weighted-sum strategy based on the prediction errors **Er** of selected sub-models;
   - 2.2 If the new data point is new FV, it is added to M_1 and then M_1 is retrained;
   - 2.3 Else
     - 2.3.1 If the new data is a changed FV, it will be used to replace the FV that makes least contribution in the recent models;
       - 2.3.1.1 If the existing FV to be replaced in M_1 is unique in the ensemble, the model M_1 before replacement is saved as a new sub-model, named M_{n+1}. The selected FV in M_1 is, then, replaced by the new data point and then M_1 is retrained;
       - 2.3.1.2 If the existing FV to be replaced in M_1 is not unique in the ensemble, no new sub-model is created and the replacement is carried out directly in M_1 and then M_1 is retrained;
     - 2.3.2 Else keep the sub-models in the ensemble unchanged.
   - 2.4 Update the prediction error **Er** of each sub-model.

Fig. 1 The main procedure of OE-FV.

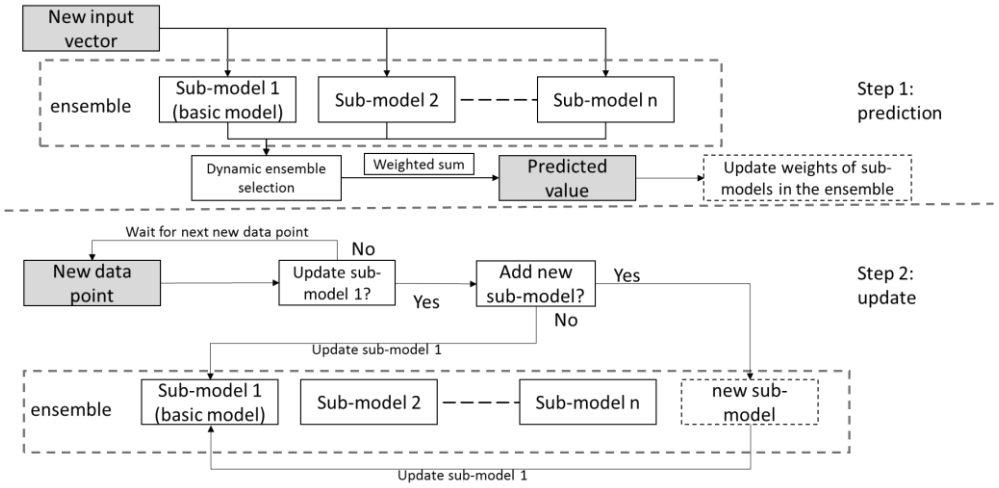

Fig. 2 Graphical updating process of OE-FV.

3.1 Training of the first sub-model in the ensemble (step 1 of Figure 1)

The first sub-model M_1 is trained on the training dataset, which is also the first and basic sub-model in the ensemble (step 1 of Figure 1). In order to reduce the model complexity and computational burden, all the training dataset are not directly used to train the first sub-model. Instead, FVS selects the representative data points, i.e. FVs, which are normally of a much smaller size than the training dataset, and M_1 is trained on the selected FVs through minimizing the MSE of the prediction on the whole training dataset. Such strategy can reduce the model complexity and keep the generalization ability of the model, at the same time. The process of FVS applied for selecting the FVs from the training dataset is shown in Appendix.

3.2 Calculation of the predicted value of a new data point (step 2.1 of Figure 1)

For any new data point coming, prediction is obtained by a dynamic ensemble selection strategy (step 2.1 of Figure 1). A dynamic ensemble selection, as presented in [29], [30] and [31], amounts to selecting the sub-models that are most relevant to the new data point and calculating their separate predictions; then, these predictions are fused by a weighted sum to give the final prediction of the ensemble for the new data point.

The dynamic selection of sub-models can be based on the overall local accuracy, local sub-model accuracy, a priori selection or a posteriori selection [29]. In OE-FV, the sub-models are selected on the basis of the local fitness of the new data point to the FVs of each sub-model, calculated by Equation (3). Only the sub-models with a local fitness that satisfies $1 - J_{Si}(\boldsymbol{x}) < \rho$ are selected to form the ensemble predictor EoC for the new data point.

Suppose $\mathbf{Er}$ is the vector that contains the cumulated prediction errors of all the sub-models and $\mathbf{Er}_{Eoc}$, which is a subset of $\mathbf{Er}$, contains the prediction errors of the sub-models in EoC: the weights of the selected sub-models are calculated as Equation (6) below:

$$\boldsymbol{\omega} = \frac{1/\boldsymbol{Er}_{Eoc}{}^2}{\sum 1/\boldsymbol{Er}_{Eoc}{}^2}, \tag{6}$$

and the prediction of the ensemble is calculated as a weighted sum of the prediction results of all the selected sub-models, as shown in Equation (7), with $\hat{y}_i$ and $\hat{y}$ being the predicted value of the *i*-th selected sub-model and of the ensemble, respectively:

$$\hat{y} = \sum_{EoC} \omega_i \hat{y}_i. \tag{7}$$

If none of the sub-models in the ensemble gives a local fitness that satisfies$1 - J_{Si}(\boldsymbol{x}) < \rho$, all the sub-models are used for calculating the prediction of the ensemble. In Equations (6), $\mathbf{Er}_{Eoc}$ is replaced by $\mathbf{Er}$ and in Equation (7), the weighed sum is carried out on all the sub-models.

3.3 Update of the ensemble with a new pattern (step 2.2 of Figure 1)

If the local fitness of the new data point to the FVs in each sub-model satisfies the relation $1 - J_{Si}(\boldsymbol{x}) > \rho$, such data point is judged as a new FV and it is added to the first sub-model M_1 that had been trained on the training dataset (step 2.2 of Figure 1). The other sub-models are not modified with the new FV, as they represent only the patterns in the data at a certain historical period and the new FV represents the ongoing pattern of the data. A new sub-model is not created in the case of a new FV, as it enriches the ensemble without decreasing its performance on the whole data. Thus, the number of sub-models are not changed and only the sub-model M_1 is updated to follow the ongoing patterns. Once the FVs in M_1 are increased by one, the model is retrained by minimizing the MSE on the recent data points (how to choose the recent data points is explained in details in Section 3.6).

3.4 Update of the ensemble with a changed pattern (step 2.3.1 in Figure 1)

When the new data point is judged as not being a new FV, the verification of a changed FV is carried out by calculating the prediction error (absolute bias between the predicted value and the true output) of all the sub-models. If the prediction errors are all bigger than a preset threshold θ, the new data point is judged as a changed pattern and is used to replace another FV in the sub-model M_1.

Before the replacement, we need to solve two questions.

The first one is how to choose the FV in M_1 to be replaced by the new data point. The pseudo-code for Online-SVR-FID in Appendix gives a way for SVR models, based on counting the times that an FV has been a support vector in the past SVR models during the adaptive learning process and, then, the contribution in the recent SVR models are more weighted than those in the older ones. Following the same strategy, a more general way is to cumulate the contribution of the FV through a weighted sum of its value calculated in Equation (5) for all the data points.

Suppose the contribution of each FV in M_1 is m_i: when a new data point is coming, Equation (5) can compute its similarity with each FV in M_1. A bigger a_i in $\boldsymbol{a}$ represents a larger similarity, and, thus, a bigger contribution to the prediction of the new data point: the contribution of the FV is, then, updated as ${m_i}^{new} = \gamma m_i + a_i$, with γ a positive value smaller than one.

Once the FV in M_1 to be replaced by the new data point is selected, the second problem is how to assure that all the past patterns are stored in the ensemble. If the selected FV is unique in the ensemble, i.e. it exists only in M_1, the replacement of this FV may cause a loss of a past pattern in the data. Thus, step 2.3.1.1 of Figure 1 proposes to "copy" the model M_1 as a new sub-model and before the replacement. The selected FV in M_1 is replaced by the new data point. With such a strategy, the changed pattern is learned by M_1 and the old pattern is not deleted from the ensemble by adding a new sub-model, which is a copy of M_1 before the replacement. Note that all the sub-models except M_1 are created this way and they can be seen as a copy of M_1 for *t* different periods. As M_1 can always follow the ongoing patterns in the data, the diversity among the sub-models represents different steps of the data stream.

If the selected FV in M_1 is not unique in the ensemble, it is replaced directly by the new data point without adding a new sub-model (step 2.3.1.2).

3.5 Update of the prediction error of sub-models (step 2.4 of Figure 1)

In Section 3.3, the sub-models weights are calculated according to their prediction errors $\mathbf{Er}$ on the data points. After the training of the first sub-model M_1 in step 1 of Figure 1, the prediction error for M_1 is the root MSE on the whole training dataset.

When a new data point is available, part of (if the new data point is not a new FV) or all (if the new data point is a new FV) the sub-models are selected to derive the prediction of the dynamically selected ensemble, as introduced in Section 3.3. In any case, sub-model M_1 is always selected, as the online learning process assures that M_1 contains all the dimensions of the available data in RKHS while the other sub-models contain only part of it. Thus, M_1 can give a local fitness for the new data point, which is smaller than or equal to those given by other sub-models. At the end of each iteration for a new data point, the strategy for updating the prediction error of the sub-models for different situations is given below:

1) For the sub-models, except M_1, in the dynamically selected ensemble SoC for the new data point $(\boldsymbol{x}_i, y_i)$, their prediction errors are updated as $\boldsymbol{Er}_{EoC} = \beta \boldsymbol{Er}_{EoC} + |\hat{\boldsymbol{y}}_i - y_i|$, with the vector $\boldsymbol{Er}_{Eoc}$ containing their prediction errors, β being a positive parameter smaller than one and $\hat{\boldsymbol{y}}_i$ the vector of predicted values of the sub-models in EoC.
2) For the sub-models that are not selected into EoC, their prediction errors are updated as $\boldsymbol{Er} = \beta \boldsymbol{Er} + \tau Er$, with Er the maximal prediction error given by the sub-models in EoC and τ a parameter bigger than one, in order to decrease the weights of these sub-models in the next iteration.
3) The case of M_1 is different from the above two types of sub-models, as it may be adaptively

updated with the new data point:

3.1) If it is not updated during steps 2.2 and 2.3 of Figure 1, its prediction error is updated as step 1.

3.2) Otherwise, it is updated with the prediction error after the update, i.e. after steps 2.2 and 2.3 of Figure 1. M_1 gives a new prediction for the new data point different from the one calculated in step 2.1 of Figure 1 during the calculation of the prediction of the ensemble for the new data point. The error of the new prediction is the true error for M_1 at the end of this iteration. Its prediction error is updated with the new prediction error as $Er_1 = \beta Er_1 + \left|\hat{y}_{1,new} - y_i\right|$, with $\hat{y}_{1,new}$ being the prediction for the new data point given by the updated M_1.

4) If a new sub-model is created during the online learning of the new data point, the prediction error of the new sub-model is calculated with $Er_{n+1} = \beta Er_1 + \left|\hat{y}_{1,old} - y_i\right|$, with $\hat{y}_{1,old}$ being the prediction for the new data point given by M_1 at step 2.1 of Figure 1, which is not updated yet with the new data point, and Er_1 being the prediction error of M_1 at the beginning of this iteration in step 2.1 in Figure 1, i.e. before updating.

3.6 Retraining of the sub-model M_1

Facing a new FV or a changed FV, the sub-model M_1 needs to be updated. However, it is not always possible to find a way to update directly the model, as shown in Online-SVR-FID without retraining it from scratch. In this paper, we suppose that M_1 is updated by retraining by minimizing the MSE on a number (much larger than the number of FVs in the model) of recent data points, in order to guarantee the generalization ability of the model. Suppose the last sub-model was added at the i_0-th data point, when the i-th data point is coming: the objective function to minimize is the MSE on the data points from i_0 to i. In order to avoid overfitting and underfitting on the recent data points, a minimal (N_{min}) and a maximal (N_{max}) number of the recent data points in the objective function are fixed during the retraining of M_1, i.e. the number of the recent data points for retraining M_1 is $\min(\max(N_{min}, i - i_0), N\_max)$.

3.7 Advantages of OE-FV

OE-FV has several advantages compared to other online learning ensemble approaches. It is an instance-based ensemble approach, which adaptively modifies the ensemble with each new data point, and, thus, can timely learn the new patterns, contrarily to data chunk-based and drift detector-based approaches for online learning ensemble. Thus, OE-FV can instantly follow the pattern drift in the data whereas the online learning ensembles based on data chunk or sliding window can only react after a sufficient number of new data points becomes available.

Also, storing all the patterns in the data makes the ensemble capable of creating new sub-models automatically when necessary, without the trouble of setting a fixed size of new data points as the data chunk-based approaches must do.

When a new sub-model needs to be created, there is no need to train this new sub-model, as it is a "copy" of the first sub-model M_1 as presented in Section 3 and the new sub-model is fixed once created. Only M_1 is updated with new data points, to follow the ongoing patterns.

The diversity of the sub-models are guaranteed, as each sub-model represents the patterns in the data during a different period, with M_1 representing the up-to-date patterns.

Finally, the new data points are all used to update the sub-models' weights, and only few of them are used to update M_1 and create new sub-models. For each new data point, instead of using all the sub-models to derive the prediction of the ensemble, only the most relevant ones are selected to form a dynamic ensemble; such strategy can reduce the computational complexity of the online learning process.

4. Experiments

In this section, comparisons of experimental results on several synthetic datasets and a real dataset are carried out mainly among Online-SVR-FID [28], Learn++.NSE [13], OWE [25] and the proposed OE-FV, considering prediction accuracy and computation time as the performance indicators.

Learn++.NSE is a typical data chunk-based approach for online learning ensemble. When a new data chunk of a fixed size N is available, a new sub-model is added if the prediction error on the new data chunk exceeds a predefined threshold ε. The sub-models' weights are updated according to their prediction error on all the data chunks, while the prediction error on the new data chunks are more weighted than those of the older ones. Learn++.NSE cannot adapt to the new patterns until a number of N new data points are available. When the ensemble is updated with the new chunk, it may not follow the ongoing patterns. There is a delay of the patterns in the ensemble compared to the patterns in the new data. And it is very difficult to decide the best size of the data chunk.

In order to solve these problems with Learn++.NSE, OWE updates the sub-models weights with the prediction error when a new data point is available. The strategy for adding a new sub-model is also different from Learn++.NSE: instead of waiting for a new data chunk, a sliding window is integrated. When a new data point is available, a window of a fixed size N moves one step ahead. When the prediction error on the data points in the window exceeds a predefined threshold ε, a new model on these data points is trained. Thus, there is no need of waiting for N new data points before adding a new sub-model. It is more flexible than Learn++.NSE, but there is still a delay compared to the instance-based approaches for online learning ensemble.

There is a pruned version of Learn++.NSE and OWE, which fixes a maximal number of sub-models. After the maximal number is reached, the old sub-model which gives worst prediction results on the new data points is deleted each time a new sub-model is added.

Section 4.1 shows the preliminary results on several synthetic datasets and more details of the comparison are given in Section 4.2. The experimental procedures are shown in Figure 3.

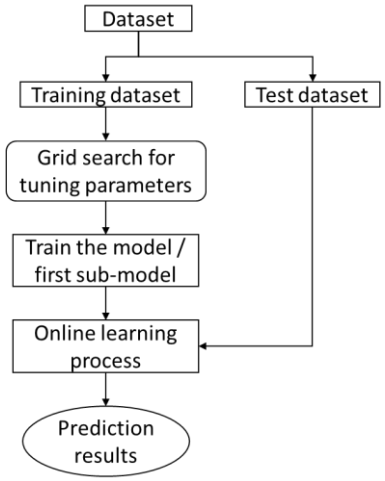

Fig. 3 Procedures for carrying out the experiments

4.1 Prediction of synthetic dataset

The proposed online learning strategy is firstly tested on a synthetic dataset with pattern drifts and recurring patterns. The synthetic data is composed of four concept and a total of $M = 30000$ data points are generated. As shown in the following equations, concepts 1 and 4 are repeated twice as recurring patterns at different period of the dataset. A Gaussian noise with a standard deviation of 0.05 is added to each input and output variable.

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

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

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

Concept 4: $y_i = (x_i^7 + x_i^8 + x_i^9)/3$, for $i = \frac{2M}{6} + 1, \dots, \frac{3M}{6}, \frac{5M}{6} + 1, \dots, M$.

The first 1000 data points form the training dataset and the rest imitate the online learning process to be fed into the model one by one.

For this dataset, one more model is added for the comparisons: the SVR with sliding window for online learning, noted as SVR in Table I. The main characteristics for computation are Inter Duo i5, 2.3 GHz, and 4G RAM.

Table I Comparisons of experimental results using Online-SVR-FID, Learn++.NSE, OWE and OE-FV.

	Online-SVR-FID	Learn++.NSE	Learn++.NSE Pruned	OWE	OWE Pruned	OE-FV	SVR
MSE	$77*10^{-4}$	$31*10^{-4}$	$15*10^{-4}$	$48*10^{-4}$	$14*10^{-4}$	$12*10^{-4}$	$21*10^{-4}$
MARE	0.1557	0.0821	0.0709	0.1062	0.0700	0.0682	0.0770
Time (s)	2613.6	18.7	20.2	352738	253.5	121.0	67090.9
# of sub-models	1	30	20	18753	20	46	1

From Table I, it is observed that Learn+.NSE Pruned, OWE Pruned and OE-FV give better results than other methods, with OE-FV gives a litter better results than the other two methods. OE-FV gives better results than SVR with much less time for online learning, as, in SVR, it takes too much time to retrain the SVR model for each new data point with a fixed number of most recent data points. While in OE-FV, the ensemble is updated only when a new / changed pattern is detected.
In order to compare statistically the prediction accuracy of OE-FV with the benchmark methods, ten synthetic drifting datasets (named separately Dataset 1, Dataset 2, …, Dataset 10) with 6000 data points are generated. Each dataset is divided into six data chunks of 1000 data points and the data of each chunk follows randomly one of the four concepts above. Recurring patterns (two non-consequent chunks follow same concept) exist in the first five datasets. For each dataset, the first 500 data points form the training dataset and the rest are in the test dataset.

Table II Prediction results (MSE and MARE) of different methods on ten synthetic datasets.

	Online-SVR-FID	Learn++.NSE	Learn++.NSE Pruned	OWE	OWE Pruned	OE-FV	SVR
	MSE						
Dataset 1	0.0065	0.0066	0.0066	0.0088	**0.0050**	0.0064	0.0064
Dataset 2	0.0075	0.0092	0.0092	0.0095	0.0064	**0.0057**	0.0084
Dataset 3	0.0079	0.0105	0.0105	0.0068	0.0070	**0.0044**	0.0096
Dataset 4	0.0019	0.0032	0.0032	0.0022	0.0025	**0.0017**	0.0032
Dataset 5	0.0029	0.0076	0.0076	0.0045	0.0048	**0.0023**	0.0061
Dataset 6	**0.0026**	0.0099	0.0099	0.0059	0.0058	0.0066	0.0073
Dataset 7	**0.0029**	0.0080	0.0080	0.0042	0.0062	0.0041	0.0076
Dataset 8	**0.0037**	0.0121	0.0121	0.0073	0.0080	0.0055	0.0098
Dataset 9	**0.0025**	0.0082	0.0082	0.0051	0.0049	0.0053	0.0062
Dataset 10	**0.0022**	0.0067	0.0067	0.0058	0.0049	0.0044	0.0060
	MARE						
Dataset 1	0.1409	0.1208	0.1208	0.1563	**0.1035**	0.1194	0.1198
Dataset 2	0.1498	0.1469	0.1469	0.1744	0.1215	**0.1193**	0.1434
Dataset 3	0.1598	0.1586	0.1586	0.1517	0.1250	**0.0154**	0.1592
Dataset 4	0.0543	0.0639	0.0639	0.0592	0.0523	**0.0474**	0.0610
Dataset 5	0.0723	0.1215	0.1215	0.0940	0.0793	**0.0713**	0.0918
Dataset 6	**0.0692**	0.1548	0.1548	0.1084	0.0841	0.0945	0.0982
Dataset 7	**0.0707**	0.1135	0.1135	0.0829	0.0842	0.0752	0.0943
Dataset 8	**0.1004**	0.2091	0.2091	0.1498	0.1283	0.1365	0.1480
Dataset 9	**0.0694**	0.1381	0.1381	0.0998	0.0783	0.0910	0.0923
Dataset 10	**0.0674**	0.0951	0.0951	0.0956	0.0718	0.0754	0.0843

Table II reports the MSE and MARE on the test dataset of each synthetic drifting dataset and the bolded values are the best results given by all the methods. It is observed that OE-FV outperforms the other methods for four out of the five datasets with recurring patterns. For a dataset without recurring patterns, the results of the ensemble approaches are influenced by the sub-models trained on the past patterns. That's why Online-SVR-FID which follows always the current patterns, gives best results on such datasets. Learn+.NSE Pruned and Learn.+NSE given same prediction results

for all datasets, as the maximum number of sub-models is set to be 20 and the non-pruned one does not pass this threshold.

Table III Rand of different methods considering their prediction accuracy on the ten synthetic datasets.

	Online-SVR-FID	Learn++.NSE	Learn++.NSE Pruned	OWE	OWE Pruned	OE-FV	SVR
	MSE						
Dataset 1	4	5.5	5.5	7	**1**	2.5	2.5
Dataset 2	3	5.5	5.5	7	2	**1**	4
Dataset 3	4	6.5	6.5	2	3	**1**	5
Dataset 4	2	6	6	3	4	**1**	6
Dataset 5	2	6.5	6.5	3	4	**1**	5
Dataset 6	**1**	6.5	6.5	3	2	4	5
Dataset 7	**1**	6.5	6.5	3	4	2	5
Dataset 8	**1**	6.5	6.5	3	4	2	5
Dataset 9	**1**	6.5	6.5	2	3	4	5
Dataset 10	**1**	6.5	6.5	4	3	2	5
	MARE						
Dataset 1	6	4.5	4.5	7	**1**	2	3
Dataset 2	6	4.5	4.5	7	2	**1**	3
Dataset 3	6	3.5	3.5	7	2	**1**	5
Dataset 4	3	6.5	6.5	4	2	**1**	5
Dataset 5	2	6.5	6.5	5	3	**1**	4
Dataset 6	**1**	6.5	6.5	5	2	3	4
Dataset 7	**1**	6.5	6.5	3	4	2	5
Dataset 8	**1**	6.5	6.5	5	2	3	4
Dataset 9	**1**	6.5	6.5	5	2	3	4
Dataset 10	**1**	5.5	5.5	7	2	3	4
Average Rank	2.400	5.975	5.975	4.600	2.600	2.025	4.425

Table III shows the ranks of different methods considering their prediction accuracy (MSE or MARE) for the same test dataset. Rank 1 means that the corresponding method gives the highest prediction accuracy.

From the average rank in the last line of Table III, it is noted that OE-FV obtained the highest rank among all the methods.

As in [36], Friedman test can check if the average ranks of the different methods are significantly different from the mean rank of all the methods under the null hypothesis, and Bonferroni-Dunn test can tell if the results of OE-FV is significantly better than those of the benchmark methods.

For k methods and N comparison results, the Friedeman statistic in Equation 8 ($R_j, j = 1, \ldots, k$ is the mean rank of method j) follows a χ_F^2 distribution with $k-1$ degrees of freedom and the statistic in Equation 9 follows the F-distribution with $k-1$ and $(k-1)(N-1)$ degrees of freedom,

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right], \quad (8)$$

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2}. \quad (9)$$

For the case studied in this paper, $k = 7$ methods on $N = 20$ comparison results are compared as shown in Table III. With the average rank in Table III for all the method, we can calculate that F_F equals to 28.3415. F_F follows the F-distribution with $7 - 1 = 6$ and $(7 - 1) \times (20 - 1) = 114$ degrees of freedom. The critical value of $F(6,114)$ for $\alpha = 0.05$ is 2.18. As the statistic F_F (28.3415) is bigger than the critical value (2.18), the null hypothesis is rejected, i.e. not all the methods perform equally for the case studies.

Bonferroni-Dunn test is used to test the significance of the difference between the average rank of two methods. The difference is significant if it is equal or bigger than the critical difference (CD) (calculated in Equation (10) with q_α the Studentized range statistic divided by $\sqrt{2}$):

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}}. \quad (10)$$

The critical value q_α of the Bonferroni-Dunn test with 7 algorithms is 2.39 for $\alpha = 0.10$. Then, the critical difference (CD) is 1.64, from Equation 10. If the average rank difference between two methods is bigger than the CD value, their performances are significantly different.

From the average ranks in Table III, we can see that, the differences among the prediction results of OE-FV, Online-SVR-FID and OWE-Pruned for the ten synthetic datasets are not significant, while they all give significantly better results than Learn+.NSE, Learn+.NSE, OWE and SVR.

4.2 Leakage prediction in NPP

In this Section, a real case study is considered, concerning a time series dataset collected from a sensor monitoring the leak flow from the first seal of a Reactor Coolant Pump (RCP) in a NPP. The function of RCP is critical for the control and safe operation of a NPP, as it pumps cold water into the reactor to evacuate the heat produced by nuclear fission.

Figure 4 is the normalized time series dataset which contains 13124 values, measured every four hours. It contains gradual, sudden and recurring data. Denoting the data as $l(t)$, the target of the work is to predict the leak flow in the next day, i.e. $y(t) = l(t + 6)$. The partial autocorrelation analysis between different time lags and the target shows that the first ten historical values are highly correlated with the target and, thus, the input vector considered is $\boldsymbol{x}(t) = [l(t - 9), l(t - 8), \dots, l(t)]$.

After the reconstruction of the original dataset, the first 500 data points form the training dataset and the rest simulate the online learning process, which feed the ensemble one by one. The basic models are all built by SVR in this experiment.

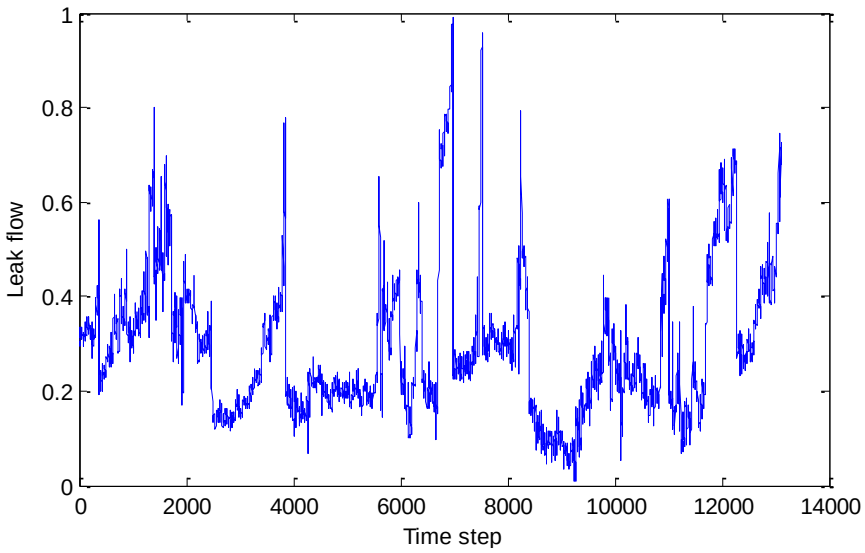

Fig. 4 Data of the leak flow in RCP.

4.2.1 Prediction results of the proposed ensemble approach for NPP

For the real case study, the different parameters in Section 3 are set as follows, based on grid search: $\rho = 10^{-6};\ \theta = 0.05;\ \gamma = 0.8;\ \beta = 0.6;\ \tau = 4;\ \mathrm{N\_min} = 150;$ and $\mathrm{N\_max} = 500$.

The online learning of a single model in [25] and OE-FV, are firstly compared in this experiment. In the case of updating a SVR model with Online-SVR-FID, a SVR model is trained on the training dataset and updated with the new data points, as proposed in [28]. In the experiment, there are totally 1198 new data points judged as changed patterns and 13 new data points as new patterns.

On the contrary, in the online learning ensemble with OE-FV, only 120 and 7 new data points are separately judged as changed and new patterns. OE-FV largely decreases the number of changed patterns and, thus, the computational complexity, as all the patterns are stored in the ensemble. Thus, OE-FV solves the problem that Online-SVR-FID has with recurring patterns. Figure 5 shows the prediction results of the test data points from 4600 to 6000 given by OE-FV and the positions of the changed and new patterns.

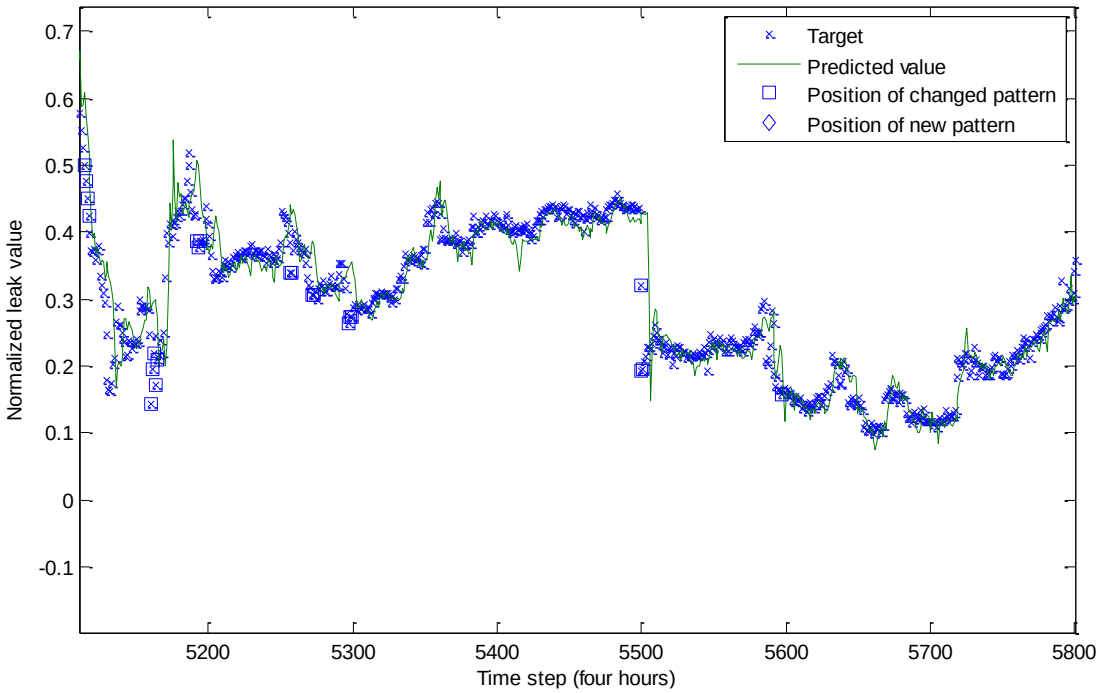

Fig. 5 Prediction results of OE-FV and the positions of changed and new patterns.

4.2.2 Results comparisons

In this case study, by grid search, the size N of the data chunk in Learn++.NSE and of the time window in OWE is fixed at 500. The threshold ε for adding a new sub-model and the discounting rate in calculating the prediction error in Learn++.NSE and OWE are (0.04, 0.2) and (0.05, 0.3) respectively. In the pruned case, the maximal number of sub-models is 20.

Table IV presents the MSE and Mean Absolute Relative Error (MARE), the computation time with the same computer (Inter Duo i5, 2.3 GHz, and 4G RAM) and the number of sub-models.

Table IV Comparisons of experimental results using Online-SVR-FID, Learn++.NSE, OWE and OE-FV.

	Online-SVR-FID	Learn++.NSE	Learn++.NSE Pruned	OWE	OWE Pruned	OE-FV
MSE	$13*10^{-4}$	$16*10^{-4}$	$16*10^{-4}$	$12*10^{-4}$	$12*10^{-4}$	$8.6*10^{-4}$
MARE	0.0977	0.1009	0.1009	0.0879	0.0882	0.0761
Time (s)	460.117	8.3607	8.0682	30485	188.394	51.299
# of sub-models	1	26	20	7513	20	13

All these approaches give comparable results considering the prediction accuracy, with Learn++.NSE giving the worst and OE-FV the best. This is caused by the update strategy integrated in the online learning ensemble. The delay during the online learning process in Learn++.NSE is longer than in OWE, and OE-FV has the shortest delay. Thus, it is verified that the instance-based approach can timely follow the ongoing patterns and give better in frequently changing environment results than data chunk-based or sliding window-based ones.

The computation burden bothering the instance-based online learning ensembles is not so obvious in OE-FV. Learn++.NSE uses least time as the ensemble is updated only when a new data chunk is available. The specific strategies proposed in OE-FV, e.g. verification of new FV and changed FV,

generation of new sub-models and dynamic ensemble selection, reduce the computational complexity of the online learning process and the results show that it uses much less time than OWE, which is based on the sliding window concept.

The time of OE-FV is also much smaller than Online-SVR-FID, as Online-SVR-FID deletes some old patterns during the updating process and when these patterns recur, it has to relearn them before giving a good prediction result. This disadvantage increases the number of updating actions during online learning and, thus, the computational burden, and decreases the prediction accuracy. On the contrary OE-FV applies a dynamic ensemble selection strategy to select the most relevant sub-models for each new data point in order to reduce the influence of the irrelevant ones. The sub-models weights are updated with each new data point and the flexibility of the ensemble is increased. In this case study, the Learn++.NSE and OWE with and without pruning give similar prediction results. As shown in [25], a larger maximal number of sub-models does not always increase the accuracy: the accuracy is no longer improved when the number of sub-models is bigger than a certain value.

Figure 6 shows the prediction results of the test data points from 5300 to 5400 given by Online-SVR-FID, Learn++.NSE, OWE and OE-FV. It is observed that OE-FV can adapt to the target faster than the others. Learn++.NSE and OWE are updated with the longest delay, as explained in the Introduction.

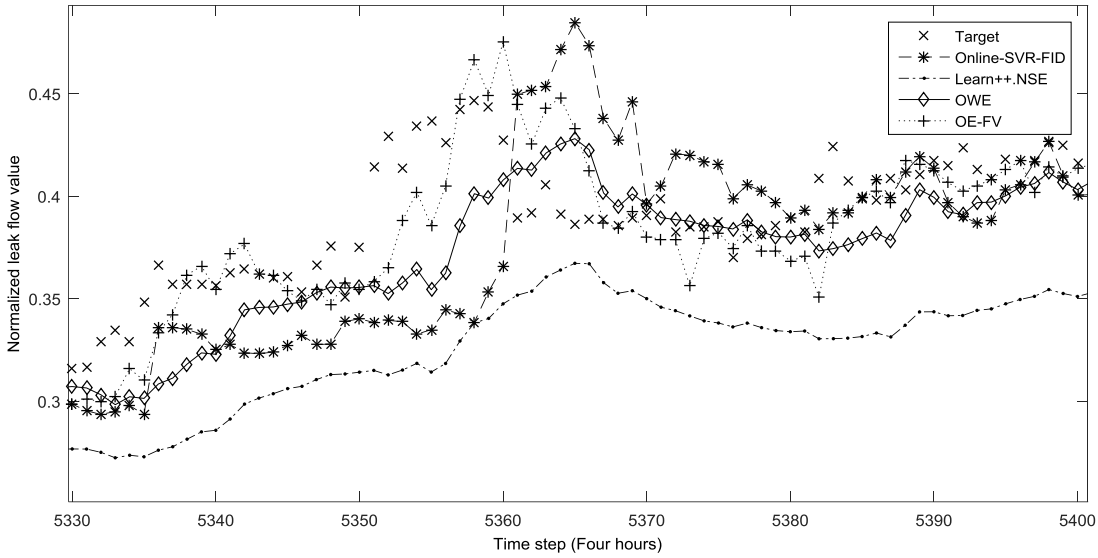

Fig. 6 Comparisons of the prediction results of the test data points from 5300 to 5400.

5. Conclusions

Based on FVS and Online-SVR-FID, a new online learning approach for SVR-based ensembles, OE-FV, is proposed. OE-FV can create an ensemble automatically from a single model. The new sub-models represent separately a certain stage of the first sub-model, whereby the diversity among them is guaranteed. To the authors' knowledge, this paradigm is used for the first time in online learning for ensembles. The dynamic ensemble selection strategy eliminates the sub-models irrelevant to the new data point, and, thus, reduces their influences on the prediction results of the ensemble. The computational burden with instance-based online learning ensemble is reduced by

taking different strategies for pattern verification and a dynamic ensemble selection.
Although developed for SVR models, the online learning strategy proposed can also be used for other kernel methods.
Comparisons on two case studies show that OE-FV outperforms Online-SVR-FID and OWE in both prediction accuracy and computation time. Learn++.NSE uses less time than OE-FV, but gives slightly worse prediction results.

Acknowledgement

The authors want to thank the author Symone Gomes Soares of [22] who shared the original code for the online learning ensemble in her paper.

References

[1] M.G. Na, S.H. Shin, S.M. Lee, D.W. Jung, S.P. Kim, J.H. Jeong, et al., Prediction of Major Transient Scenarios for Severe Accidents of Nuclear Power Plants, IEEE Trans. Nucl. Sci. 51 (2004) 313–321. doi:10.1109/TNS.2004.825090.
[2] W.J. Kim, S.H. Chang, B.H. Lee, Application of neural networks to signal prediction in nuclear power plant, 1993. doi:10.1109/23.234547.
[3] K. Kim, E.B. Bartlett, error prediction for a nuclear-power-plant fault-diagnostic adviser using neural networks, Nucl. Technol. 108 (1994) 283–297.
[4] C. Alippi. Intelligence for Embedded Systems. Springer, 2014.
[5] E.W.M. Lee, An incremental adaptive neural network model for online noisy data regression and its application to compartment fire studies, Appl. Soft Comput. 11 (2011) 827–836. doi:10.1016/j.asoc.2010.01.002.
[6] 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.
[7] L. Csató, M. Opper, Sparse on-line gaussian processes., Neural Comput. 14 (2002) 641–668. doi:10.1162/089976602317250933.
[8] N. Oza, S. Russell, Online ensemble learning, Aaai/Iaai. 6837 (2000) 1109–1109. doi:10.1007/978-3-642-22763-9_7.
[9] M.D. Muhlbaier, R. Polikar, An ensemble approach for incremental learning in nonstationary environments, in: Proc. 7th Int. Conf. Mult. Classif. Syst., 2007: pp. 490–500. http://www.springerlink.com/index/T146H82X602221UR.pdf.
[10] L.L. Minku, X. Yao, DDD: A new ensemble approach for dealing with concept drift, IEEE Trans. Knowl. Data Eng. 24 (2012) 619–633. doi:10.1109/TKDE.2011.58.
[11] Z. Ouyang, M. Zhou, T. Wang, Q. Wu, Mining Concept-Drifting and Noisy Data Streams Using Ensemble Classifiers, in: 2009 Int. Conf. Artif. Intell. Comput. Intell., 2009: pp. 360–364. doi:10.1109/AICI.2009.153.
[12] W.N. Street, Y. Kim, A streaming ensemble algorithm (SEA) for large-scale classification, Proc. Seventh ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. - KDD '01. 4 (2001) 377–382. doi:10.1145/502512.502568.

[13] M.D. Muhlbaier, R. Polikar, An ensemble approach for incremental learning in nonstationary environments, in: Proc. 7th Int. Conf. Mult. Classif. Syst., 2007: pp. 490–500. http://www.springerlink.com/index/T146H82X602221UR.pdf.
[14] Z. Erdem, R. Polikar, F. Gurgen, N. Yumusak, F. Gürgen, N. Yumuşak, Ensemble of SVMs for Incremental Learning, Simulation. 3541 (2005) 246–256. doi:doi: 10.1007/11494683_25.
[15] D. Brzezinski, J. Stefanowski, Reacting to different types of concept drift: the Accuracy Updated Ensemble algorithm., IEEE Trans. Neural Networks Learn. Syst. 25 (2014) 81–94. doi:10.1109/TNNLS.2013.2251352.
[16] R. Razavi-Far, P. Baraldi, E. Zio, Dynamic Weighting Ensembles for Incremental Learning and Diagnosing New Concept Class Faults in Nuclear Power Systems, Nucl. Sci. IEEE Trans. 59 (2012) 2520–2530. doi:10.1109/TNS.2012.2209125.
[17] K. Nishida, K. Yamauchi, O. Takashi, ACE: Adaptive Classifiers-Ensemble system for concept-drifting environments, in: Mult. Classif. Syst., 2005: pp. 176–185. http://dx.doi.org/10.1007/11494683_18.
[18] L.I. Kuncheva, Classifier Ensembles for Changing Environments, in: Mult. Classif. Syst., 2004: pp. 1–15. doi:10.1007/978-3-540-25966-4_1.
[19] L.L. Minku, A.P. White, X. Yao, The impact of diversity on online ensemble learning in the presence of concept drift, IEEE Trans. Knowl. Data Eng. 22 (2010) 730–742. doi:10.1109/TKDE.2009.156.
[20] E. Page, Continuous inspection schemes, Biometrika. 41 (1954) 100–115. doi:10.2307/2333009.
[21] M. Baena-Garcia, J. del Campo-Avila, R. Fidalgo, A. Bifet, R. Gavalda, R. Morales-Bueno, Early Drift Detection Method, in: 4th ECML PKDD Int. Work. Knowl. Discov. from Data Streams, 2006: pp. 77–86.
[22] I. Zliobaite, A. Bifet, B. Pfahringer, G. Holmes, Active learning with drifting streaming data, IEEE Trans. Neural Networks Learn. Syst. 25 (2014) 27–39. doi:10.1109/TNNLS.2012.2236570.
[23] J.Z. Kolter, M. a Maloof, Using additive expert ensembles to cope with concept drift, Int. Conf. Mach. Learn. 119 (2005) 449–456. doi:10.1145/1102351.1102408.
[24] P. Kadlec, B. Gabrys, Adaptive local learning soft sensor for inferential control support, in: 2008 Int. Conf. Comput. Intell. Model. Control Autom. CIMCA 2008, 2008: pp. 243–248. doi:10.1109/CIMCA.2008.66.
[25] 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.
[26] P. Li, X. Wu, X. Hu, Mining Recurring Concept Drifts with Limited Labeled Streaming Data, ACM Trans. Intell. Syst. Technol. 3 (2012) 1–32. doi:10.1145/2089094.2089105.
[27] G. Baudat, F. Anouar, Feature vector selection and projection using kernels, Neurocomputing. 55 (2003) 21–38. doi:10.1016/S0925-2312(03)00429-6.
[28] J. Liu, E. Zio, An Adaptive Online Learning Approach for Support Vector Regression, Mechanical System and Signal Processing. (submitted).
[29] A.H.R. Ko, R. Sabourin, A.S. Britto, From dynamic classifier selection to dynamic ensemble selection, Pattern Recognit. 41 (2008) 1735–1748. doi:10.1016/j.patcog.2007.10.015.
[30] P.R. Cavalin, R. Sabourin, C.Y. Suen, Dynamic selection approaches for multiple classifier systems, Neural Comput. Appl. 22 (2011) 673–688. doi:10.1007/s00521-011-0737-9.
[31] M. Alrifai, T. Risse, Combining global optimization with local selection for efficient QoS-

aware service composition, Proc. 18th Int. Conf. World Wide Web - WWW '09. (2009) 881. doi:10.1145/1526709.1526828.
[32] G. Cauwenberghs, T. Poggio, Incremental and Decremental Support Vector Machine Learning, Learning. 13 (2001) 409. doi:10.1.1.21.1720.
[33] A.E. Hoerl, R.W. Kennard, Ridge Regression: Biased Estimation for Nonorthogonal Problems, Technometrics. 12 (1970) 55–67. doi:10.1080/00401706.1970.10488634.
[34] A.J. Smola, B. Sch, B. Schölkopf, A Tutorial on Support Vector Regression, Stat. Comput. 14 (2004) 199–222. doi:10.1023/B:STCO.0000035301.49549.88.
[35] M. Seeger, Gaussian processes for machine learning., 2004. doi:10.1142/S0129065704001899.
[36] J. Demšar, Statistical Comparisons of Classifiers over Multiple Data Sets, J. Mach. Learn. Res. 7 (2006) 1–30. doi:10.1016/j.jecp.2010.03.005.

Appendix

Online-SVR-FID is divided into two parts, i.e. Offline Training and Online Learning. The Offline Training contains the feature vector selection and SVR model training, while the Online Learning contains the actions to take, when facing a new or a changed pattern.

The pseudo-code of Online-SVR-FID is repeated below.

Initialization:
Training dataset: $\boldsymbol{T}_r = \{(\boldsymbol{x}_i, y_i)\}$, for $i = 1, 2, \dots, M$
Testing dataset: $\boldsymbol{T}_e = \{(\boldsymbol{x}_i, y_i)\}$, for $i = M+1, M+2, \dots, M+H$
Feature space: $\mathbf{S}$ = [ ]
Threshold of local fitness: ρ
Threshold of prediction error: θ

Offline Training:
First FV in $\mathbf{S}$:
For *i* = 1 to M calculate
$\mathbf{S} = \{\boldsymbol{x}_i\}$, compute global fitness $J_{\boldsymbol{S}}$.
End for.
Select the point which gives the maximum of the global fitness as the first FV and add it to $\mathbf{S}$ as the first FV.
$\boldsymbol{T}_r$ is reduced as the complement of $\mathbf{S}$ in $\boldsymbol{T}_r$, i.e. $\boldsymbol{T}_r = \boldsymbol{T}_r \backslash \boldsymbol{S}$.
Second and the other FVs:
Calculate local fitness for data points in $\boldsymbol{T}_r$ with the present feature space $\mathbf{S}$;
Select the data point *k* which gives the minimum of local fitness;
If $1 - J_{\boldsymbol{S},k} > \rho$, this point is a new FV and added to $\mathbf{S}$;
$\mathbf{E} = \{(\boldsymbol{x}_k, y_k)\ and\ (\boldsymbol{x}_i, y_i) : 1 - J_{\boldsymbol{S}}(\boldsymbol{x}_i) \leq \rho\}$ and $\boldsymbol{T}_r$ is reduced as the complement of $\mathbf{E}$ in $\boldsymbol{T}_r$, i.e. $\boldsymbol{T}_r = \boldsymbol{T}_r \backslash \mathbf{E}$;
If $1 - J_{\boldsymbol{S},k} \leqslant \rho$, end the process of FVs selection;

Train the SVR model on the FVs in S to minimize the MSE on the whole training dataset.

Online Learning:

When a new data point $(\boldsymbol{x}_N, y_N)$ is available

DO

Calculate the local fitness $J_{\boldsymbol{S},N}$ of this new data point;

If $1 - J_{\boldsymbol{S},N} > \rho$

ADDITION: this new data point is a new FV; add it to S and add this new data point in the model using the Incremental Learning [32]. Go back to the beginning of Online learning and wait for the next new data point.

If $1 - J_{\boldsymbol{S},N} \leqslant \rho$, verify the bias between the target of this new data point and the predicted value

If the bias is smaller than θ

Keep the model unchanged. Go back to the beginning of Online learning and wait for the next new data point.

Otherwise

UPDATE: find the FV with least contribution for the SVR models and nonzero value in Eq. (5). Unlearn this FV found with decremental learning [32] and add the new data point with incremental learning. Go back to the beginning of Online learning and wait for the next new data point.

Strategy for selecting the least contribution FV to be updated in UPDATE.

1. A vector $\boldsymbol{m} = (m_1, m_2, \ldots, m_l)$ 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. Otherwise, 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 $\boldsymbol{a}$ for the new data point according to Equation (5). Then, among all the FVs in the model with non-zero values in $\boldsymbol{a}$, the one with least contribution, say m_I, is deleted from the model using Decremental Learning as in [29] 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 [29] 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.