Failure prognostics by support vector regression of time series data under stationary/nonstationary environmental and operational conditions

Jie Liu

ECOLE CENTRALE DES ARTS ET MANUFACTURES «ECOLE CENTRALE PARIS»

THESE

Présenté par

JIE LIU

Pour l'obtention du

GRADE DE DOCTEUR

Spécialité: Génie Industriel

Laboratoire d'accueil: Laboratoire Génie Industriel

SUJET:

Failure Prognostics by Support Vector Regression of Time Series Data under Stationary/Nonstationary Environmental and Operational Conditions

Soutenue le: 12 Février 2015

Devant un jury composé de:

Cesare ALIPPI	**Rapporteur**
Noureddine ZERHOUNI	**Rapporteur**
Christian DERQUENNE	**Examinateur**
Pierre DERSIN	**Examinateur**
Enrico ZIO	**Examinateur & Directeur de thèse**

N° ordre: 2015ECAP0019

To my family

ACKNOWLEDGEMENTS

I would like to express my profound gratitude and deep regards to my supervisor of this thesis, Professor Enrico Zio, for his exemplary guidance, monitoring and constant encouragement throughout the course of this thesis. The blessing, help and guidance given by him time to time shall carry me a long way in the journey of life on which I am about to embark. It was a pleasure and a great honor to pursue a Ph.D. degree under his supervision.

I would like to express my sincere appreciation and gratitude to my co-advisor, Dr. Valeria Vitelli, for her support, time and patience. This research work would have been much more difficult without her help.

My deepest appreciation goes to all the jury members that agreed to be part of the committee: Professor Cesare Alippi, Professor Noureddine Zerhouni, Dr. Pierre Dersin and Dr. Christian Derquenne. I am particularly grateful to the reviewers, Professor Cesare Alippi and Professor Noureddine Zerhouni, for their careful reading of the manuscript and all the constructive and helpful remarks, which helped to improve the quality of my thesis work.

I would like to thank the China Scholarship Council for its support, allowing me to conduct this three-year Ph.D. work under the Chair on Systems Science and the Energy Challenges (SSEC).

I would like to offer my special thanks to my family for their love, support and constant encouragement. My parents are the meaning of my life, and I undoubtedly could not have done this without them.

I would like also to take this opportunity to thank all my friends/colleagues in SSEC and at Laboratory of Industrial Engineering, Ecole Centrale Paris. It has been a pleasure to work with them during the last three years. I want to give them my great thanks for everything they have done, for their encouragement and their disposition to offer me a hand whenever I needed help. In addition, great thanks to my office mates, Muxia Sun, Yanhui Lin and Ronay Ak for being at my side.

Last but not least, special thanks to the director of the Laboratory of Industrial Engineering, Ecole Centrale Paris, Professor Jean-Claude Bocquet, for a three-year hosting.

ABSTRACT

This Ph.D. work is motivated by the possibility of monitoring the conditions of components of energy systems for their extended and safe use, under proper practice of operation and adequate policies of maintenance. The aim is to develop a Support Vector Regression (SVR)-based framework for predicting time series data under stationary/nonstationary environmental and operational conditions. Single SVR and SVR-based ensemble approaches are developed to tackle the prediction problem based on both small and large datasets. Strategies are proposed for adaptively updating the single SVR and SVR-based ensemble models in the existence of pattern drifts. Comparisons with other online learning approaches for kernel-based modelling are provided with reference to time series data from a critical component in Nuclear Power Plants (NPPs) provided by Electricité de France (EDF). The results show that the proposed approaches achieve comparable prediction results, considering the Mean Squared Error (MSE) and Mean Relative Error (MRE), in much less computation time.

Furthermore, by analyzing the geometrical meaning of the Feature Vector Selection (FVS) method proposed in the literature, a novel geometrically interpretable kernel method, named Reduced Rank Kernel Ridge Regression-II (RRKRR-II), is proposed to describe the linear relations between a predicted value and the predicted values of the Feature Vectors (FVs) selected by FVS. Comparisons with several kernel methods on a number of public datasets prove the good prediction accuracy and the easy-of-tuning of the hyperparameters of RRKRR-II.

Key words: Feature selection, Model interpretability, Nuclear power plant, Online learning, Prediction, Ensemble, Support vector regression, Time series data

RESUME

Ce travail de thèse est motivée par la possibilité de surveiller l'état des composants de systèmes d'énergie pour leur utilisation prolongée et sécuritaire, conformément à la pratique correcte de fonctionnement et des politiques adéquates de maintenance. La motivation est de développer des méthodes basées sur la régression à vecteurs de support pour la prédiction de données de séries chronologiques dans des conditions environnementales et opérationnelles stationnaire / non stationnaire. Les simples modèles et les ensembles de modèles à base de SVR sont développés pour attaquer la prédiction basée sur des petits et des grands ensembles de données. Des stratégies sont proposées pour la mise à jour de façon adaptative les simples modèles et les ensembles de modèles à base de SVR au cas du changement de la distribution générant les données. Les comparaisons avec d'autres méthodes d'apprentissage en ligne sont fournies en référence à des données de séries chronologiques d'un composant critique dans les centrales nucléaires fournis par Electricité de France (EDF). Les résultats montrent que les approches proposées permettent d'atteindre des résultats de prédiction comparables compte tenu de l'erreur quadratique moyenne et erreur relative, en beaucoup moins de temps de calcul.

Par ailleurs, en analysant le sens géométrique de la méthode de la sélection de vecteurs caractéristiques (FVS) proposé dans la littérature, une nouvelle méthode géométriquement interprétable, nommé Reduced Rank Kernel Ridge Regression-II (RRKRR-II), est proposée pour décrire les relations linéaires entre un valeur prédite et les valeurs prédites des vecteurs caractéristiques sélectionné par FVS. Les comparaisons avec plusieurs méthodes sur un certain nombre de données publics prouvent la bonne précision de la prédiction et le réglage facile des hyperparamètres de RRKRR-II.

Mots Clés: Apprentissage en ligne, Centrale nucléaire, Ensemble, Interprétation du modèle, Les données de séries chronologiques, Prédiction, Régression à vecteur de support, Sélection des vecteurs caractéristiques

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

PART I: GENERALITIES

This part includes the first two Chapters of this thesis which presents the context, relevance and the selected base method for the research work.

1. INTRODUCTION

The research presented in this thesis concerns the development of a Support Vector Machine (SVR)-based framework for failure prognostics of components in energy systems, in particular in Nuclear Power Plants (NPPs). The present introductory chapter of the thesis is structured as follows. Section 1.1 discusses failure prognostics and positions it within maintenance engineering. Section 1.2 reviews the methods for failure prognostics for components of NPPs. Section 1.3 presents the component object of the reference case study throughout the thesis: the first seal in the Reactor Coolant Pump (RCP) of a NPP. Section 1.4 motivates the choice of the basic method, i.e. SVR, for performing failure prognostics. Section 1.5 states the research motivations and objectives. Section 1.6 presents the structure of the thesis.

1.1 Failure prognostics

In modern industry, the demand for high reliability of systems, low environmental risks, and assurance of human safety during operating processes, has substantially increased in the last decades. The timely maintenance of a System, Structure or Component (SSC) is critical for the profitability and competitiveness of industrial companies. The economic loss of an unexpected shutdown may cost a company up to hundreds of thousands of Euros [53]. The economic loss of shutting down a NPP in USA is $1.25 million/day [96]. In the current competitive marketplace, maintenance management and machine health monitoring play a more and more important role in gaining market share by reducing equipment downtime, associated costs and scheduling disruptions. For all the above, the development of effective strategies for the maintenance of SSC is a strong motivation for companies and public works.

Different strategies for SSC maintenance have been developed in the years, whereby the relatively recent Condition-Based Maintenance (CBM) aims at maintaining the SSC in operating conditions, monitoring its state based on sensors measurements, including event data and condition monitoring data [97]. Failure prognostics, based on failure time prediction and Remaining Useful Life (RUL) estimation, has become a major focus in the research and development of CBM in various technological fields, such as aerospace, automotive, nuclear, and national defense [117].

1.1.1 Failure prognostics

Prognostic, originated from the Latin word *prognosticus* and from the Greek word *prognōstikos*,

is an adjective that relates to prediction or foretelling and a noun for a sign or symptom indicating the future course of a disease or sign or forecast of some future occurrence, as defined in the American Heritage Dictionary [33].

Various technical definitions of prognostics have been given within the maintenance engineering discipline. According to ISO:13381-1, prognostics is defined as the "estimation of time to failure and risk for one or more existing and future failure modes"[58], [59], [164], [157]. In [118] and [163], prognostics is the process of predicting the future health of the SSC of interest, based on the current and historical health conditions.

In this thesis, prognostics is intended as an overarching concept which includes the capability to provide early detection and isolation of precursor and/or incipient fault conditions to a SSC failure and to have the technology and means to manage and predict the progression of this fault condition to a SSC failure, as defined in [31], [83] and [165]. This definition of prognostics contains two objectives: short-term prediction and RUL prediction.

Various classifications of the prognostic approaches have been proposed in the literature [71], [117], [134]. The most useful one attempts to distinguish the different approaches according to the type of information and data they use. Hence, prognostic approaches can be divided into four categories: Model-based or Physics of Failure-based approaches, Knowledge-based approaches, Data-driven approaches, and Combination approaches [117].

1.1.2 Characteristics of failure prognostic approaches

The development of a prognostic approach may face quite different situations with respect to the information and data available on the past, present and future behavior of the SSC. There might be situations in which sufficient and relevant data on the SSC behavior are available, others in which the SSC behavior is known well enough to build a sufficiently accurate model, and yet others in which scarce data concerning the SSC are available (e.g. due to the fact that, being the equipment highly valued, it has very low degradation and failure rates), but in which process and functional data related to the SSC degradation and failure processes are measured by specific sensors. Correspondingly, a wide range of approaches, based on different sources of information, modeling schemes and data processing algorithms has been developed [178].

According to [178], any prognostic approach should have desirable characteristics of:

- **Robustness**: the performance of the prognostic method does not degrade abruptly in the presence of noise, uncertainties or unexpected situations.

- **Uncertainty quantification**: estimates and predictions are accompanied by a measure of the associated error, accounting for the incomplete and imprecise information available on the process.
- **Adaptability**: the prognostic approach is able to take into account the changes in the environment and in the SSC of interest, and to perform well in different operating conditions.
- **Generalization power**: the prognostic approach can be used for prognostics of both complex and relatively simple systems, as well as for components of a system.

In the next section, we will briefly review the different types of prognostic approaches and compare them according to the previous four characteristics.

1.1.3 Types of failure prognostic approaches

Model-based approaches usually employ physical-mathematical models to describe the physical processes having direct or indirect effects on the health of the SSC under study [50], [63], [65], [85], [89], [112]. Physical-mathematical models require specific theoretic and mechanistic knowledge relevant to the SSC of interest and are normally developed by domain experts. The parameters in the models are calibrated on data. Statistical techniques are used to define thresholds to detect and identify the presence of faults.

Knowledge-based approaches are based on expert systems, which embed the expert knowledge and its reasoning manipulation for inference of the solution to the particular prognostic problem. It combines the power of computers with the laws of reasoning on expertise. Knowledge-based approaches store the domain knowledge extracted by human experts into computers in the form of rules, and then use these rules to generate solutions. Expert systems [10], [14], [82] and fuzzy logic [172] are two typical examples of knowledge-based approaches [117].

Data-driven approaches statistically and probabilistically predict the future health of the SSC of interest, based on historical and current data related to the degradation state of the SSC [134], [45]. Data-driven approaches tackle the direct estimation and prediction of the degradation state without modeling the equipment physical behavior and operation. Such approaches include conventional statistical algorithms, like linear regression [106] and time series analysis [46], as well as machine learning and data mining algorithms, like Artificial Neural Networks (ANNs) [176], and Support Vector Machines (SVMs) [51].

Combination approaches (also called ensemble-based approaches) attempt to take advantage of

the strength of different data-driven approaches or both data-driven and model-based approaches, by fusing information from different approaches [6], [116], [121], [181]. Compared with single models, ensemble-based approaches achieve higher accuracy by aggregating results from different sub-models, higher adaptability by training diverse sub-models and higher robustness to noise [6].

Table I compares these four types of prognostic approaches with respect to the characteristics introduced in Section 1.1.2 (the more stars (*) are assigned to a prognostic approach for a specific characteristic, the better it performs in that specific characteristic).

Table I. Comparisons of different prognostic approaches.

Characteristic / Approach	Robustness	Uncertainty quantification	Adaptability	Generalization power	Others
Model-based	YES *** (with respect to the noise in the data)	YES	YES ** (by parameter re-calibration)	NO * (a model is built for a specific type of SSC)	It is only applicable in situations where accurate mathematical models can be constructed from first principles.
Knowledge-based	YES ** (with respect to the noise in the data, but not to the unknown situations)	YES	YES ** (by rule base modification)	NO * (a model is built on the knowledge of a specific SSC.)	In complex cases it is difficult to obtain domain knowledge and convert it to rules. When the number of rules increases dramatically, a combinatorial explosion problem related to conditions checking may cause dramatic computational burden.
Data-driven	YES ** (dependent on the training dataset)	YES	YES *** (by retraining)	YES ***	The performance of data-driven approaches are highly-dependent on the quantity and quality of the data for training and validation.
Combination	YES ***	YES	YES *** (by retraining and/or addition in the ensemble)	YES ***	Ensemble methods have some disadvantages: the increased storage, the increased computational burden, the decreased comprehensibility, etc.

1.2 Failure prognostics of NPP components

The case study of reference in the thesis regards a critical component of a NPP, whose

monitoring is crucial to guarantee the NPP safe operation.

There are currently more than 400 NPPs in the commercial global fleet around the world, and another 222 projects are in various stages of development [11]. The global demand for electricity continues to grow [11]. The problem of meeting the growing energy demand has to cope with the requirement, arising in most countries, to minimize carbon emissions through the use of carbon-free electricity generation. In this global scenario, nuclear power projects for electricity generation will play a crucial role in the long-term management of energy sources, and nuclear energy can be expected to remain a significant part of the global energy mix [88].

In existing NPPs, the most crucial tasks to be accomplished are improving safety, maintaining availability and reducing operation and maintenance costs. Moreover, extensions in power up rates and in the average component life duration increase the need for techniques for diagnosing and predicting the NPPs health, because the occurrence of component degradation and failure becomes more and more likely as load is increased or changed, and as age advances [88].

In the future, with more than 550 NPPs under construction or plan with new technology and design-for-inspectability concept, failure prognostics is also very critical to know the state of these new complex system [96].

Some works have already appeared in the relevant literature concerning prognostics of NPP components. In [179] and [180], a fuzzy similarity analysis is introduced to find a combination of the reference patterns, weighed by their similarity to the observed failure pattern, to determine the future evolution of the observed failure pattern and to derive the corresponding RUL. In [104] and [174], SVM is used to predict the collapse moment for wall-thinned pipe bends and elbows. In [78], a systematic approach is introduced for the prediction of pump performance characteristics, for situations in which the experimental data are not available. In [74], the authors present a framework for the control of the steam generator water level in the secondary circuit of a NPP, based on an extension of the standard linear model predictive control algorithm to linear parameter varying systems. A back propagation ANN is proposed for the prediction of thermal power in NPPs in [129]. In [105], a probabilistic neural network is applied for classifying accidents into groups of initiating events and a fuzzy neural network is used to identify the major severe accident scenarios after the initiating events. Back propagation networks are used in [155] to estimate one or more process variables by establishing the nonlinear relationship among a set of plant variables. In [69], an ANN is used to estimate the value of the undetected next-time-step signal of the steam generator water level in a NPP. In [144], data from plant operation experience are used in combination with in-service inspections

and degradation management programs to ensure that the degradation mechanisms do not adversely impact plant safety.

A lot work in the literature use data-driven approaches for the prognostics of components of NPPs [105]. Model-based approaches are less suitable due to the complexity of NPP systems, because the physical relations among different variables are difficult to establish, especially for the reference case study in this thesis. Knowledge-based and ensemble approaches are also used for the prognostics of NPP components; however, since critical components rarely fail during operation, the related experience can be limited even for the experts in the field. Thus, in such cases, data-driven approaches are the best choice.

1.3 The reference case study: leakage in the first seal of the reactor coolant pump

Pumps play a major role in the safe operation of NPPs. Their operating characteristics play a significant role in determining the thermal and hydraulic behavior of nuclear reactors in following transients. The Reactor Coolant Pump (RCP) is a critical component of a NPP, since it guarantees enough cold water in the reactor vessel to protect the nuclear materials and to deliver the heat released from the nuclear fission to the steam generator.

Figure 1 shows the position of RCP in a NPP and the structure of a RCP is shown in Figure 2.

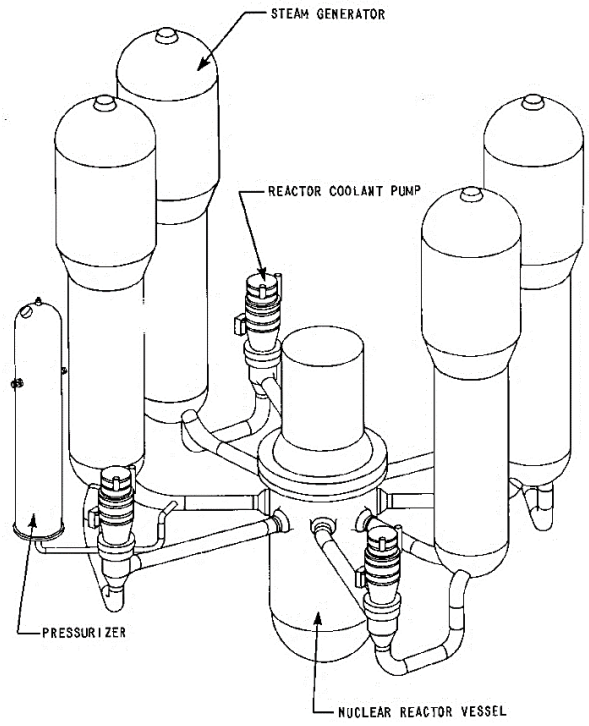

Fig. 1 Simplified NPP system (http://www.nucleartourist.com/images/rcs.gif).

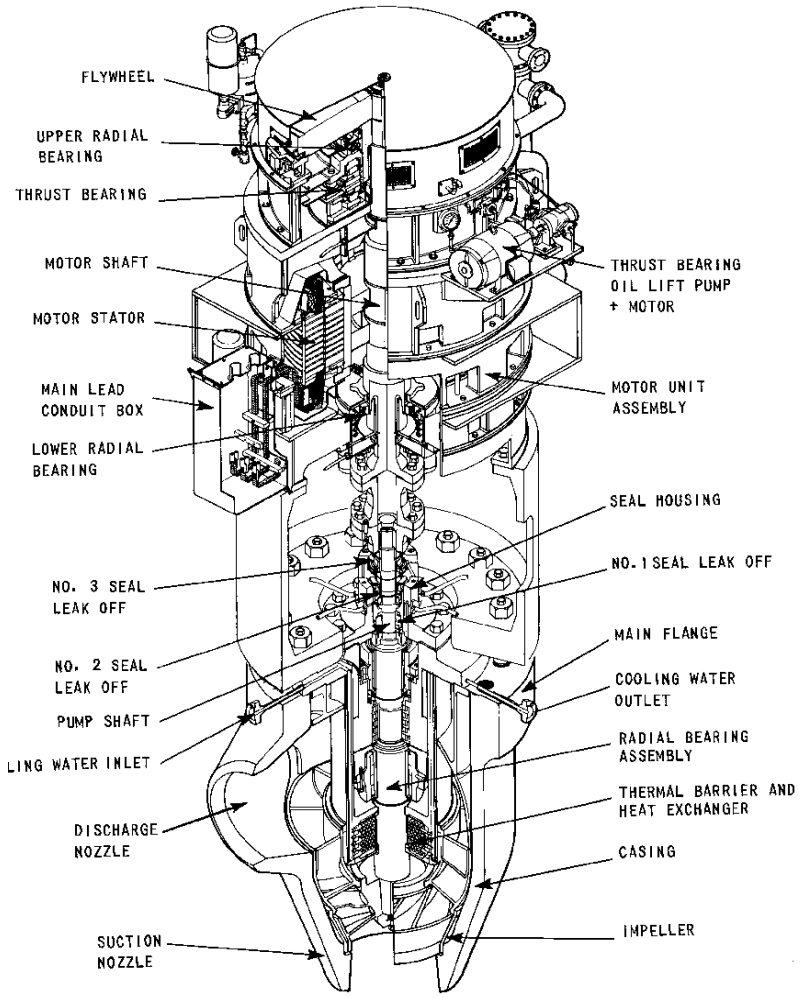

Fig. 2 Structure of a RCP in NPP (http://www.nucleartourist.com/systems/rcs1.htm).

The RCP is composed of three main parts: the pump part which includes pump casing, guide vanes, guide shaft, main flange, impeller, heat shield and so on; the sealing part which includes bearing seal system, electrical motivation support base, couplings; the motor part which includes upper and lower bearings, upper and lower machine frame, stator, rotor, flywheel, oil lifting systems, air coolers and oil coolers [187].

The sealing system is composed of three seals, which are sequentially named the first seal, the second seal and the third seal. The sealing system prevents the boron water from leaking outside the primary circuit of a NPP. The leaked boron water may endanger the personnel working in the NPP and the equipment inside the nuclear island. Thus, the reliable control of the leak flow is very important. If the leak flow exceeds a predefined threshold, the NPP should be shut down to decrease the chance of severe disasters.

The reference case study in this thesis concerns the failure prognostics of the leak flow of the

first seal. For this, 20 failure scenarios from 10 NPPs have been considered. For each NPP, the leak flow was monitored every four hours, but starting from different time instances and for different durations; hence, the number of measurements in the time series is different for each NPP. The fault has occurred at different times, and in some scenarios the operators managed to bring the pump back to a normal condition. Table II shows the number of transients in each scenario.

Table II Number of transients in each scenario.

Scenario	1	2	3	4-6	7-10	11	12-14	15	16-17	18-19	20
# of time series data points	2277	385	385	2017	1391	3124	562	964	2767	1061	861

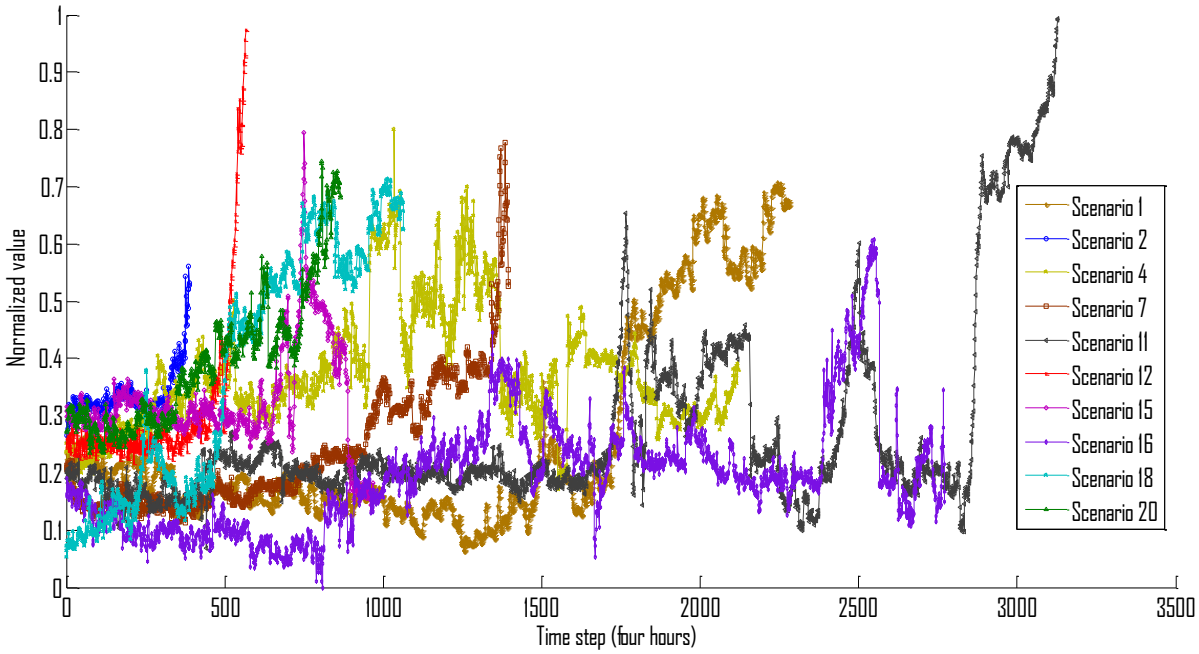

Fig. 3 Normalized data of some available scenarios.

1.4 Failure prognostics with data-driven approaches

Considering the complexity of the component of interest and the scarce knowledge on the causes of failure, data-driven approaches have been chosen to tackle the failure prognostic problem in this thesis.

A large number of data-driven approaches have been developed based on Artificial Intelligence

(AI) techniques, such as perceptron-based approaches, kernel-based approaches, Markov chain models, hazard rate approaches and other methods. In this Section, two popular and promising data-driven approaches are reviewed, i.e. perceptron-based approaches and kernel-based approaches. A kernel-based approach, Support Vector Machine (SVM) is chosen for the work of this thesis, and various approaches based on it are developed for failure prognostics under different situation requirements.

1.4.1 Perceptron-based approaches: ANNs

The perceptron-based approaches predict the output of a new input vector by calculating the weighted sum of the outputs from all the processing elements (modes) of the perceptron. The perceptron-based approaches are trained on the training data points by running the algorithm repeatedly until it finds the predictions which are (approximately) correct on all the training data points [76]. The method that best represents this kind of approaches is ANN.

ANN is a data processing system which can be used to estimate the nonlinear regression function describing the relationship between a set of inputs and outputs, and this estimation is achieved through a network training procedure. The network structure is composed by three types of layers: input layer, hidden layer (sometimes more than one) and output layer. Each layer has a number of simple, neuron-like processing elements called “nodes” or “neurons” that interact with each other by using numerically weighted connections [153]. The structure of a classical ANN (feed-forward ANN) is shown in Figure 4.

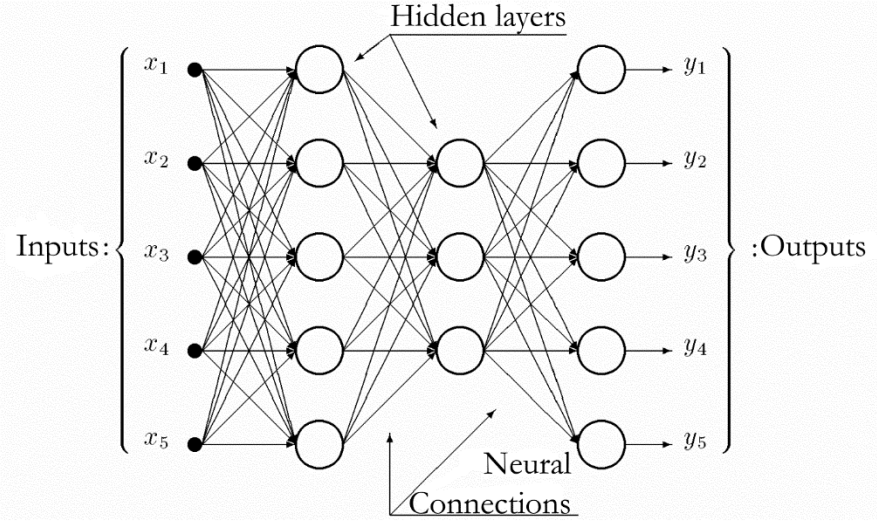

Fig. 4 Structure of a feed-forward ANN with two hidden layers (http://bulyaki.com/2012/11/04/feedforward-neural-networks/).

There are several types of neural networks, e.g., Dynamic Wavelet Neural Networks (DWNN) [166], Polynomial Neural Networks (PNN) [111], Weighted Probabilistic Neural Networks (WPNN) [143] and Self-Organization Maps Neural Networks (SOMNN) [81]. Some applications of ANN in prediction problems can be found in references [36], [56], [68], [90], [154] and [182].

Choosing the optimal number of hidden neurons is a big challenge for using ANN-based approaches, as an underestimate of the number of neurons can lead to poor approximation and generalization power, while an overestimate of the number of neurons can result in overfitting and eventually make the search for the global optimum more difficult [17], [73]. The computational burden for dealing with irrelevant features is also a big drawback of ANN-based approaches.

1.4.2 Kernel-based approaches: SVMs

In the last decades, benefiting from the computational simplicity and the good generalization performance in statistical machine learning problems, kernel-based machine learning methods have drawn much attention for regression [88], [98], [100], classification [30], [80], [130] and unsupervised learning [136], [137], [138]. Good and comprehensive reviews of these methods can be found in [52] and [99]. Support Vector Machine (SVM) [1], [18], [141], Kernel Gaussian Process (KGP) [44], [125], [169], Kernel Ridge Regression (KRR) [49], [57], [133], Kernel Logistic Regression (KLR) [77], [186], Kernel Principal Component Analysis (KPCA) [139], [173] are some of the most popular kernel methods.

The nonparametric and semi-parametric representer theorems given in [135] show that for a large class of kernel algorithms with Structural Risk Minimization (SRM), i.e., minimizing a sum of a structural risk term and a regularization term, in a Reproducing Kernel Hilbert Space (RKHS), the optimal solutions can be written as a kernel expansion supported on training data points. The estimate function of the kernel methods, including SVM, KGP, KRR, KLR and KPCA, can all be formulated as

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

where $f(\boldsymbol{x})$ is the estimate function describing the relation between the data points $(\boldsymbol{x}_i, y_i), i = 1,2,\dots,N$; $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the inner product of the mapping of $\boldsymbol{x}_i, \boldsymbol{x}_j$ in RKHS; α_i are the weights to optimize and b is a constant that can be zero or non-zero. Note that the unknowns in Equation (1) have no practical meanings. Note also that, normally, in kernel

methods there are three types of hyperparameters: the penalty factor C, which is a trade-off between the empirical risk term and the regularization term; the hyperparameters related to the definition of the empirical risk term (e.g. the parameter ϵ in the ϵ-insensitive loss function of SVM); the hyperparameters related to the kernel function itself (e.g. the parameter σ in the Gaussian Radial Basis kernel Function (RBF) written as $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 / 2\sigma^2}$). SVM is a typical kernel-based approach and it is called Support Vector Regression (SVR) when used for regression and prediction problems.

The training optimization problem of kernel methods necessarily reaches a global minimum, and avoids falling in a local minimum, which may happen for other methods, such as ANN. However, the main drawbacks of kernel methods are the unacceptable computational burden for training with large datasets, the difficulty in tuning the hyperparameters and the lack of interpretability of the mechanisms within the model.

1.4.3 Comparison between ANN and SVM

ANN and SVM are the two most popular data-driven approaches used for supervised learning, being respectively a perceptron-based approach and a kernel-based approach, and they can both be applied for the failure prognostics problem tackled in this thesis.

There has been some comparisons reported in the literature between ANN and SVM on different case studies. In [62], the authors compare different machine learning methods, including ANN and SVM for chemical toxicity classification with/without filter-based feature selection. ANN and SVM give comparable prediction results and outperform the other machine learning methods in all the tests. In [158], the performances of feed-forward ANN and SVM are compared in the prediction of traffic speed based on real time data and historic data, collected by various systems in transportation networks. Considering the mean absolute percentage error on the test dataset, SVM performs better than ANN with less training data points and the performance of an ANN model is highly dependent on the size of the training dataset. It is also shown that the capability of a SVM model is less dependent on the training dataset. In [96], a comparison is carried out between ANN and SVM for ECG arrhythmias classification. The Kernel-Adatron (K-A) learning algorithm is integrated with SVR and the experiment proves that SVM is much faster than backpropagation ANN in training, while the Mean Squared Error (MSE) on the test dataset is higher than the one of ANN. In [84], an empirical comparison is carried out between ANN and SVM for face recognition with binary

and multi-class classification problems. The results show that they give comparable results in binary classification, while SVM greatly outperformed ANN for multi-class problems. The comparison results between ANN and SVM in [151] prove again the superiority of SVM in both training speed and results. Finally, SVM outperforms ANN in the work of [2], [4], [21], [22], [95] and [114].

It is not possible to have a strong and conclusive statement about the superiority of either SVM or ANN in supervised machine learning. These two methods are comparable on robustness, adaptability and generalization power, and both may suffer the high computational burden and the demand for large enough dataset, problems that are shared by all data-driven approaches. The reasons that lead us to select SVM (SVR) in this thesis are listed below:

1. SVM is a non-parametric method, which involves sound theory first, then implementation and experiments [145].
2. The solution to an SVM is global and unique, i.e. less prone to overfitting, since SVM uses the SRM [156].
3. SVM has a simple geometric interpretation, gives a sparse solution and its computational complexity is not dependent on the dimensionality of the input space [147].
4. SVM demands less training data points and is more efficient in the regression problem with small size of training datasets [128].

1.5 Research objectives

Prediction is the basic and most important part of prognostics for CBM. Accurate and timely prediction of the health conditions of the SSC of interest can provide useful information for rational maintenance planning, to gain high production, safety benefits, human, environment and asset protection.

The reference case study in this thesis consists in predicting the leak flow of the first seal in the RCP, for which a physical model is hard to build but there are enough measurements available for regression and prediction. Following the discussion in previous sections, SVR is selected to predict the leak flow in the first seal. In the rest of this thesis, SVR is used in place of SVM, as the problem tackled in this thesis focuses on regression and prediction.

Many efforts of the research on SVR have been devoted to studying situations in which a sufficiently large and representative dataset is available from a fixed, albeit unknown

distribution. Correspondingly, the first objective in this thesis is to develop SVR-based approaches for regression and prediction on the reference case study in a static environment. The model trained for these situations can function well for patterns within the representative training dataset [122].

On the other hand, in real-world applications, the SSC of interest may be operated in nonstationary environments and evolving operational conditions, whereby patterns drifts. Then, in these situations, to be of practical use the constructed models must be capable of timely learning changes in the existing patterns and new patterns arising in the dynamic environment of SSC operation. In the reference case study, different NPPs may be operated in different environments and there maybe pattern drifts among different failure scenarios or even in different parts of the same scenario. Thus, the second objective of this thesis is to provide robust SVR-based approaches with a good generalization ability in nonstationary environments.

The research context can be further divided into four different situations: single SVR model for small dataset without pattern drifts; SVR-based ensembles for large dataset without pattern drifts; adaptive single SVR model for small dataset with pattern drifts and online learning ensembles for large dataset with pattern drifts.

The time horizon of prediction can be divided into long term (larger than 1 month) and short term (multiple time steps). Long-term prediction provides information for maintenance planning, while short-term prediction is for taking emergency actions, i.e. shutting down the NPP. In this thesis, the time horizon is fixed as one-day ahead prediction (6 steps ahead) after the discussion with experts in EDF.

1.6 Structure of the thesis

The thesis is composed of four main parts. Part I (Chapters 1-2) introduces the different types of prognostic approaches, the failure prognostics of components in NPP and the undertaken research objectives. It illustrates the specific data-driven approach chosen for these objectives, i.e. SVR, and the challenges for applying SVR for regression and prediction. Part II (Chapters 3-9) illustrates the methods developed and applied in this Ph.D. work, discusses briefly the results obtained in the case studies, and provides general conclusions and some future works. Part III lists the references cited in Part I and Part II. Part IV is a collection of six selected papers, scientifically reporting the outcomes of the research work, which the readers are referred to for further details.

For what concerns Part I, the Introduction Chapter summarizes the different approaches for failure prognostics and presents the details of the reference case study, the objectives of this thesis and the organization of the thesis. According to the characteristics of the reference case study, after comparisons between SVM and ANN, SVM (SVR) is chosen as the basic approach for failure prognostics in this thesis. Chapter 2 reviews the challenges of the application of SVR and some solutions that have been proposed in the literature.

Part II gives details of the approaches developed in this thesis under different situations introduced in Section 1.5, and it briefly reports application results on the case study of interest. Precisely, Chapter 3 (Paper I) describes the single Probabilistic Support Vector Regression (PSVR) model for prediction with small datasets and without pattern drifts, with an effective innovative strategy proposed for tuning hyperparameters. Experimental results show the efficiency and accuracy of the proposed strategy on the case study. A strategy for training a SVR model with selected FVs is also proposed in this Chapter. In order to guarantee the generalization ability, the model is built on the selected Feature Vectors (FVs) and aims at minimizing the Mean Squared Error (MSE) on the whole training dataset. Chapter 4 (Papers II & III) describes the proposed dynamic weighted ensemble approaches for prediction with large datasets and without pattern drifts. Instead of fixing the weights of the sub-models during training and testing, the weight of each sub-model for each data point is calculated on the basis of the fuzzy similarity presented in [179] and local fitness in Feature Vector Selection (FVS) [5]. The comparisons with the single PSVR model and the fixed-weighted ensemble show the superiority of the proposed dynamic weighting strategy on the case study. The computational burden brought by the dynamic weights calculations are acceptable considering the much better results. In Chapter 5 (Paper IV), online learning strategies are proposed for a single SVR model with a small dataset, and under nonstationary environmental and operational conditions, i.e. pattern drifts. Two types of pattern drifts are defined in this Chapter and a criterion is proposed for the online learning strategy to identify each type of pattern drifts. Different actions are undertaken to update the model according to the detected type of pattern drifts. An adaptive online learning ensemble is proposed in Chapter 6 (Paper V) to build an ensemble and update the weights of sub-models automatically, in the context of a large dataset prediction problem. Finally, by analyzing the geometrical properties of the data points in RKHS, a novel kernel method, named Reduced Rank Kernel Ridge Regression-II (RRKRR-II) is proposed in Chapter 7 (Paper VI). RRKRR-II describes the linear relation between a predicted value and those of the selected FVs. Comparisons of the experimental results with various popular kernel methods

on five public datasets show that RRKRR-II gives always comparable prediction accuracy with the best results given by all the benchmark methods on the case study. In Chapter 8, some conclusions on the original contributions and perspectives are drawn.

Figure 5 gives a pictorial view of the thesis structure, the research background, the objective of the thesis and the methodological approaches considered in the present work.

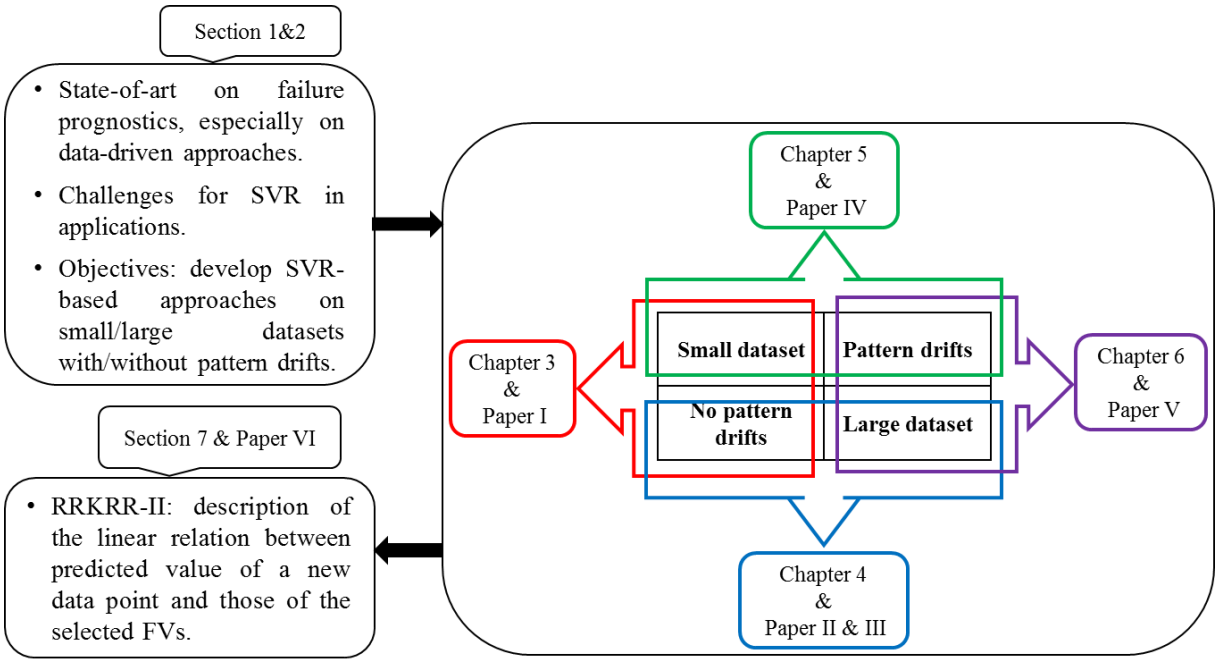

Fig. 5 A pictorial view of the structure of this thesis.

2. CHALLENGES OF APPLYING SVR TO TIME SERIES DATA FOR FAILURE PROGNOSTICS

The objective of this thesis is to develop SVR-based approaches for stationary and nonstationary environments for the leakage of first seal in RCP. SVR is a supervised learning technique from the field of machine learning applicable to regression. Rooted in the book "Statistical Learning Theory" and developed by Vladimir Vapnik and co-workers at AT&T Bell Laboratories in 1995, SVR is based on the principle of SRM [161].

An important feature of SVR is that the solution is based only on those data points which are at the margin. These points are called Support Vectors (SVs). The linear SVR can be extended to handle nonlinear problems when the data is first transformed into a high dimensional feature space, i.e. RKHS, using a set of nonlinear basis functions. In RKHS, the data points can be expressed by a linear estimate function. An important advantage of SVR is that it is not necessary to implement this transformation and to determine the linear estimate function in RKHS: a kernel representation can be used instead, and then the solution is written as a weighted sum of the values of certain kernel functions evaluated at the SVs. According to the SRM principle [162], used in the construction of an SVR, the generalization error rate is upper bounded by a formula containing the training error and the Vapnik-Chervonenkis (VC) dimension, which describes the capacity of the model.

2.1 Basics of SVR

Suppose $\boldsymbol{T} = \{(\boldsymbol{x}_i, y_i): i = 1,\ 2, \dots, N\}$ is the training dataset. SVR finds a function $f(\boldsymbol{x}) = \boldsymbol{\omega x} + b$ that has at most ε deviation from the actually obtained targets y_i for all the training data points, and at the same time is as flat as possible. In other words, we do not care about the errors as long as they are less than ε, but will not accept any deviation larger than this. However, this may not always be the case, meaning that we also want to allow some error larger than ε. Analogously to the soft margin loss function [8], which is adapted to SVR in [24], one can introduce slack variables ξ_i, ξ_i^* in the constraints of the optimization problem of SVR.

For simplicity, we first introduce the linear SVR, whose associated optimization problem is

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

$$subject\ to \begin{cases} y_i - f(\boldsymbol{x}) \le \varepsilon + \xi_i \\ f(\boldsymbol{x}) - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \\ f(\boldsymbol{x}) = \boldsymbol{\omega x} + b \end{cases}. \quad (2)$$

The constant C determines the trade-off between the flatness of $f(\boldsymbol{x})$ and the amount up to which deviations larger than ε are tolerated. This corresponds to dealing with an ε-insensitive loss function which is shown in Figure 6 [87]. Only the data points outside the shaded region contribute to the cost. It turns out that in most cases the optimization problem in Equation (2) can be solved more easily in its dual formulation.

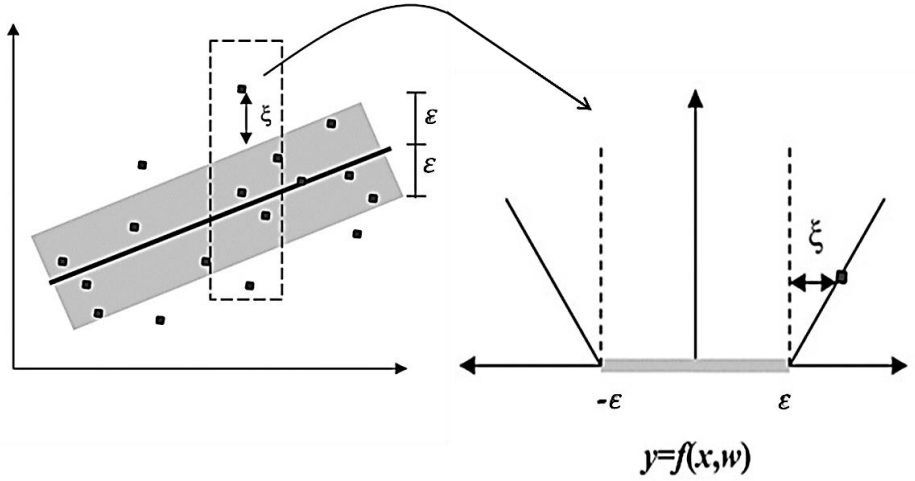

Fig. 6 The soft margin loss setting for SVR [87].

The dual optimization problem of Equation (2) is

$$L = \frac{1}{2}\|\boldsymbol{\omega}\|^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) - \sum_{i=1}^{N}(\eta_i\xi_i + \eta_i^*\xi_i^*) - \sum_{i=1}^{N}\alpha_i(\varepsilon + \xi_i - y_i + \boldsymbol{\omega x_i} + b) - \sum_{i=1}^{N}\alpha_i^*(\varepsilon + \xi_i^* + y_i - \boldsymbol{\omega x_i} - b). \quad (3)$$

Here L is the Lagrangian and η_i, η_i^*, α_i, α_i^* are positive Lagrange multipliers. By substituting the partial derivatives of L with respect to the primal variables, i.e. $\boldsymbol{\omega}, b, \xi_i, \xi_i^*$, into Equation (3), the dual optimization problem becomes

$$maximize\ \frac{1}{2}\sum_{i,j=1}^{N}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\boldsymbol{x}_i\boldsymbol{x}_j - \varepsilon\sum_{i=1}^{N}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{N} y_i(\alpha_i - \alpha_i^*)$$

$$subject\ to\ \sum_{i=1}^{N}(\alpha_i - \alpha_i^*) = 0\ and\ \alpha_i, \alpha_i^* \in [0, C]. \quad (4)$$

The partial derivative of L with respect to the primal variable $\boldsymbol{\omega}$ shows that $f(\boldsymbol{x})$ can be rewritten as

$$f(\boldsymbol{x}) = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)\boldsymbol{x}_i\boldsymbol{x} + b. \quad (5)$$

Equation (5) is the so-called support vector expansion, i.e. $\boldsymbol{\omega}$ can be described as a linear combination of the training data points. In such sense, the complexity of a function's representation of SVR is independent of the dimensionality of the input space, and depends only on the number of SVs.

The linear SVR predictor described in Equation (5) solves the linear regression problem. In the case of nonlinear regression, the training data inputs are mapped into a high-dimensional feature space, i.e. RKHS, by mapping $\boldsymbol{\varphi}$ as described in [109]. Then, the standard linear SVR predictor described above is applied to the data points in the feature space, where the mapping of the training data points allows describing their complex relationship as a linear one.

In SVR, we do not need to know explicitly the mapping $\boldsymbol{\varphi}$. By introducing the kernel product describing the inner product of two inputs in the feature space, i.e. $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\varphi}(\boldsymbol{x}_i)\,\boldsymbol{\varphi}(\boldsymbol{x}_j)$, the optimization problem in Equation (4) can be rewritten as

$$maximize\ \frac{1}{2}\sum_{i,j=1}^{N}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\,k(\boldsymbol{x}_i, \boldsymbol{x}_j) - \varepsilon\sum_{i=1}^{N}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{N} y_i(\alpha_i - \alpha_i^*)$$

$$subject\ to\ \sum_{i=1}^{N}(\alpha_i - \alpha_i^*) = 0\ and\ \alpha_i, \alpha_i^* \in [0, C], \tag{6}$$

and the expansion of $f(\boldsymbol{x})$ can be rewritten as

$$f(\boldsymbol{x}) = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)k(\boldsymbol{x}_i, \boldsymbol{x}) + b. \tag{7}$$

SVR has been widely used in different domains with promising results, such as inverse geosounding problem [43], seismic liquefaction potential [39], geo- and environmental sciences [37], [92], [123], protein fold and remote homology detection [124], image retrieval [148], facial expression classification [42], end-depth and discharge prediction in semi-circular and circular shaped channels [27], traffic speed and travel time prediction [26], breast cancer prognosis [91], underground cable temperature prediction [47], etc. But there are still some challenges for applying SVR approaches, including high computational complexity with large datasets, tuning of hyperparameters (e.g. C, ε and parameters related to the kernel function) and adaptive learning in nonstationary environments.

2.2 Reducing computational complexity

The computational complexity of training a SVR model increases exponentially with the size of the training dataset [48], [110]. For a large dataset, the computation time may become inacceptable. In the literature, there are three directions to reduce the computational complexity.

The first direction consists in dividing the training dataset into sub-datasets, and then training an ensemble of SVR models. The main advantage of the ensemble approach is to increase the generalization ability of the model, and thus to decrease the risk of overfitting [20], [55], [159], [160].

The second direction consists in reducing the computational complexity related to the solving process of the optimization problem. In [34], the authors show that for a low-rank kernel matrix it is possible to design a better interior point method in terms of storage requirements as well as computational complexity. They, then, suggest an efficient use of a known factorization technique to approximate a given kernel matrix by a low rank matrix, which in turn will be used to feed the optimizer. Several approaches are proposed to build a solution by solving a sequence of small scale problems: popular examples are stochastic gradient ascent algorithms such as the Kernel-Adatron [75] and the Sequential Minimal Optimization (SMO) [119], and active set methods, such as Chunking [12], Decomposition [113] and Shrinking [60]. The chunking algorithm starts with an arbitrary subset (chunk of data, working set) which can fit in the memory and solves the optimization problem on it by the general optimizer. SVs remain in the chunk, while other points are discarded and replaced by a new working set with gross violations of Karush-Kuhn-Tucker (KKT) conditions. SMO takes the decomposition idea to an extreme and optimizes a subset of two points at each iteration. The power of SMO derives from the fact that no extra optimization package is required, since an analytical solution for a two-point optimization problem can always be given explicitly. Dong et al. [29] introduce a parallel optimization step to quickly remove most of the non-support vectors, where block diagonal matrices are used to approximate the original kernel matrix so that the original problem can be split into hundreds of sub-problems which can be solved more efficiently. In addition, some effective strategies such as kernel caching and efficient computation of kernel matrix are integrated to speed up the training process. SVM^{light} [61] is a general decomposition algorithm, where a good working set is selected by finding the steepest feasible direction of descent with q nonzero elements. The q variables that correspond to these elements compose the working set.

The last direction for reducing the computational burden associated to SVR consists in reducing the size of the training dataset by preprocessing. Some approaches are based on the characteristics of the input vector in RKHS, e.g. KPCA [171], Feature Vector Selection (FVS) [5], convex Hull vertices selection [54], Orthogonal Least Squares (OLS) regression [23], Minimum Enclosing Ball (MEB) [152], Sparse Online Gaussian Process (SOGP) [16] etc.

Other methods are based on maximizing the prediction accuracy, e.g. orthogonal least squares learning algorithm [23], Fisher Discriminant Analysis [140], significant vector learning [35], kernel F-score feature selection [120], etc.

2.3 Tuning of hyperparameters

Hyperparameters play a critical role in the prediction performance of a SVR model. Many approaches have been proposed for tuning hyperparameters, but none of the proposed approaches is efficient and accurate in all applications. In [19], Analytic Parameter Selection (APS) is proposed to calculate the hyperparameters values directly from the training dataset. But it is shown that a combination of APS and Genetic Algorithm (GA) can give better prediction results [183]. In [86] and [185], a Particle Swarm Optimization (PSO)-based approach for parameter determination and feature selection for SVR is proposed, and the method has the advantage of no loss of the prediction accuracy. By maximizing the evidence in the Bayesian-based SVR, one can systematically tune hyperparameters and select input features, while the evidence gradients are expressed as the averages over the associated posterior and can be approximated using Hybrid Monte Carlo (HMC) sampling [41]. The novel method proposed in [184] uses Particle Filtering (PF) to estimate the hyperparameters according to the whole measurement sequence up to the last observation instance. By treating the SVR model as the observation equation of a particle filter, this method allows updating the hyperparameters dynamically when a new observation comes.

2.4 SVR adaptive online learning

Many 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. In real-world applications, the SSC of interest is usually operated in nonstationary and evolving environmental and operational conditions. Then, to be of practical use, models must be capable of timely adapting to changes in the existing patterns, and of learning newly arising patterns in the dynamic environment of SSC operation. These approaches for nonstationay environment can be divided into adaptive single models and online learning ensembles.

Some online learning approaches for single SVR models have been proposed in the literature, for SVR to adaptively learn new data patterns. In these approaches, the online learning of a trained SVR model is mostly based on the prediction accuracy and/or characteristics of the input

vectors of the data points. The strategy is to add data points as basis for the SVR model when they are not predicted well and/or contain new information on the input space. In [107], the authors propose a novel approach based on an adaptive KPCA and SVR for real-time fault diagnosis of High-Voltage Circuit Breakers (HVCBs). In [167], the authors propose an online core vector machine classifier with adaptive MEB adjustment. In [16], the authors combine a Bayesian online algorithm with a sequential construction of relevant subsets of the training dataset and they propose a Sparse On-line Gaussian Process (SOGP) approach to overcome the limitation of Gaussian processes on large datasets. The methods above consider only the characteristics of the inputs to update the model, not the prediction accuracy. In [15], the authors propose an online recursive algorithm to "adiabatically" add or remove one data point in the model while retaining the KKT conditions on all the other data points in the model. In [64], the authors propose a multiple incremental algorithm for SVR, based on the previous results. These incremental and decremental learning approaches feed to the model all new points, including noisy and useless ones, without selecting the most informative ones. In [25], the authors propose online passive-aggressive algorithms for classification and regression, but the method considers only the prediction accuracy as the update criterion. In [67], the authors use classical stochastic gradient descent within the feature space, and some straightforward manipulations, for online learning with kernels. The gradient descent method destroys completely the KKT conditions, which instead are necessary for building a SVR model.

Online learning ensemble approaches are also effective strategies for tackling a nonstationary environment. There are different types of approaches for online learning ensembles, e.g. data-chunk-based approaches, drift detector-based approaches, instance-based approaches, etc. Accuracy Weighted Ensemble (AWE) is proposed in [170] to train a new classifier on each new incoming data chunk and to update the sub-models' weights according to their accuracy on the past and present data chunks. The Streaming Ensemble Algorithm [146] builds separate sub-models on sequential data chunks, which are then combined into a fixed-size ensemble using a heuristic replacement strategy. Learning++.NSE [101] trains a new sub-model on the new data chunk if the prediction error exceeds a predefined threshold, and combines it with previous sub-models through a dynamically modified weighted majority voting. The sub-models' weights are calculated with their weighted-sum performance on different data chunks. All approaches mentioned so far train new sub-models on the new data chunk. Similar approaches are also used in [13], [32] and [175]. The problem with these data chunk-based approaches is the determination of the size of the data chunk: bigger chunks give more stable sub-model

estimation but different drifts may be contained in one sub-model. On the other hand, smaller chunks can better separate different drifts but lead to worse sub-model estimation. 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 may no longer be the ongoing ones.

In order to overcome these difficulties, various online learning ensemble approaches are proposed in the literature, which may combine a drift detector with online learning ensemble to alarm the need for a new sub-model, or update the ensemble with each single data point. Adaptive Classifier Ensemble (ACE) [108] slowly builds a new sub-model when the sub-models' error with the new data reaches a certain threshold. In [126], pattern drifts are detected by measuring the normalized weighted average output of the sub-models in the ensemble. Diversity analysis is used in [94] to divide different drifts. The most popular drift detector algorithm is the Drift Detection Method (DDM) [115], which models the prediction error on each data point according to a binominal distribution. A modified version of DDM, called EDDM, is proposed in [3], and it gives better results but is more sensitive to noise. A new approach for online learning ensemble, called Diversity for Dealing with Drift (DDD) is proposed in [93], and it manages to maintain ensembles with different diversity levels. The experimental results show that DDD gives robust and more 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 [177], a theoretically supported framework for active learning of drifts in data streams is presented, and three active learning strategies are developed based respectively on uncertainty, dynamic allocation of labeling efforts over time and randomization of search space. AddExp in [72] adapts sub-models' weights according to their actual losses in terms of prediction error, and a decreasing factor is integrated to reduce the weights of sub-models which perform poorly. The Incremental Local Learning Soft Sensing Algorithm (ILLSA) [66] is also an instance-based approach which contains two parts: one is based on training different sub-models on data points from different patterns; the other part consists in updating the sub-models' weights for each new data point, according to the posterior probability given by a Bayesian framework. Another instance-based approach, named Online Weighted Ensemble (OWE), is proposed in [142] 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 associated to updating the ensemble with every new data point. A dynamically weighted ensemble is proposed in [40] to store only the most relevant features to the learnt concept, an approach which in turn increases the memory efficiency.

The previous three challenges are inevitable when developing SVR-based prognostic approaches. In this thesis, possible solutions for these three challenges are proposed for SVR-based single model or ensemble approaches.

PART II: RESEARCH DEVELOPMENT

This part is the main body of the thesis which includes 6 Chapters (from Chapter 3 to Chapter 8) and presents the original contributions of the research work.

3. SINGLE SVR MODEL FOR FAILURE PROGNOSTICS

In this Chapter, we consider the simplest situation with a small dataset, and without pattern drifts. Strategies for the preprocessing of the time series dataset and for tuning hyperparameters are proposed. The details of the experimental results on the reference case study of this thesis are reported in Paper I of Part IV. In this section, we first introduce the Bayesian version of SVR with error bar estimation presented in [87], followed by the proposed efficient strategy for tuning hyperparameters. Then, a modified version of the FVS introduced in [5] is presented and, then, the strategy for training a SVR model on the FVs selected from the training dataset is presented, aiming at reducing the computational complexity and at the same time, keeping the robustness of the model.

3.1 Probabilistic Support Vector Regression (PSVR) for failure prognostics

3.1.1 Basics of PSVR

Bayesian probabilistic paradigm has been considered in combination with SVR [102], [103], [170]. Recently, it has been shown that SVRs can be interpreted as a Maximum A Posteriori (MAP) solution to a Bayesian inference problem with Gaussian priors and an appropriate likelihood function. The method using MAP for SVR estimation is called Probabilistic Support Vector Regression (PSVR). Bayesian approaches for SVR allow obtaining an error bar along with the prediction [87].

Let us assume that the input data is a n-dimensional set of vectors $\boldsymbol{X} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \dots, \boldsymbol{x_N}\}$, independently drawn in $\boldsymbol{R}^p$, and that we also have an independent sample from the target value $\boldsymbol{Y} = \{y_1, y_2, \dots, y_N\}$, where $y_i \in \boldsymbol{R}, i = 1, 2, \dots, N$.

In regression methods, the final aim is to find an underlying function $f(\boldsymbol{x})$: $\boldsymbol{R}^p \rightarrow \boldsymbol{R}$ describing the relation between the input data and the target. We now briefly state the PSVR approach for the estimation of $f(\boldsymbol{x})$; further mathematical details on the derivation of the method can be found in the Appendix of Paper I, and in the references therein.

We make the following assumptions:

(1) Training data set $\boldsymbol{\Gamma} = \{\boldsymbol{X}, \boldsymbol{Y}\}$ follows an identical and independent distribution (i.i.d).

(2) The *a priori* probability distribution is $P[\boldsymbol{f}(\boldsymbol{X})] \propto \exp(-\frac{1}{2} \| \hat{P} f \|^2)$, where $\| \hat{P} f \|^2$ is a positive semi-definite operator and $\boldsymbol{f}(\boldsymbol{X}) = (f(\boldsymbol{x_1}), f(\boldsymbol{x_2}), \dots, f(\boldsymbol{x_N}))^T$.

(3) The ε-insensitive loss function is chosen as the loss function.

(4) The covariance function is $K(\boldsymbol{x}, \boldsymbol{x}')$, and $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\frac{|x_i - x_j|^2}{2\gamma^2})$, where $\boldsymbol{x}_i$, $\boldsymbol{x}_j$ are the input data points in $\boldsymbol{X}$.

The a posteriori probability of $\boldsymbol{f}(\boldsymbol{X})$ can be written as

$$P[\boldsymbol{f}(\boldsymbol{X})|\boldsymbol{\Gamma}] = \frac{[G(C,\varepsilon)]^N}{\sqrt{det 2\pi K_{X,X} P[\boldsymbol{\Gamma}]}} \exp\{-C \sum_{\boldsymbol{x}_i \in \boldsymbol{X}} L_\varepsilon\big(y_i - f(\boldsymbol{x}_i)\big) - \frac{1}{2}\boldsymbol{f}(\boldsymbol{X})^T K_{\boldsymbol{X},\boldsymbol{X}}^{-1} \boldsymbol{f}(X)\}, \quad (8)$$

where $G(C, \varepsilon) = \frac{1}{2}\frac{C}{C\varepsilon+1}$, $K_{\boldsymbol{X},\boldsymbol{X}} = [K(\boldsymbol{x}_i, \boldsymbol{x}_j)]$ is the covariance matrix of the data points of $\boldsymbol{X}$ and $L_\varepsilon(x)$ is the ε-insensitive loss function.

We find the maximum of Equation (8) using the so-called MAP. This requires finding the minimum of the following function

$$R_{GSVM}(a) = C \sum_{\boldsymbol{x}_i \in \boldsymbol{X}} L_\varepsilon\big(y_i - a(\boldsymbol{x}_i)\big) + \frac{1}{2}\boldsymbol{a}(\boldsymbol{X})^T K_{\boldsymbol{X},\boldsymbol{X}}^{-1} \boldsymbol{a}(\boldsymbol{X}) \quad (9)$$

We can see that the risk of Gaussian SVR is equivalent to the standard SVR in Section 2.1. Following the discussion in [9], [38] and [102], [149], we can write the solution of the minimization problem associated to Equation (9) in the form of Equation (7).

3.1.2 Error bar estimation

In a Bayesian treatment of the prediction problem, error bars arise naturally from the predictive distribution. They are made up of two terms, one due to the *a posteriori* uncertainty (the uncertainty of $f(\boldsymbol{x})$), and the other due to the intrinsic noise in the data [87]. Suppose that $\boldsymbol{x}$ is a test input vector, and that the corresponding value of the target is the random variable y, obtained adding to $f(\boldsymbol{x})$ an unknown noise δ with zero mean; then

$$P[\boldsymbol{\Gamma}|\boldsymbol{f}(\boldsymbol{X})] \propto \exp(-C \sum_{i=1}^{N} l(\delta_i)). \quad (10)$$

We can also obtain the density of the noise δ

$$P[\delta] = \frac{C}{2(C\varepsilon+1)} \exp(-C l_\varepsilon(\delta)), \quad (11)$$

and the noise variance

$$\sigma_\delta^2 = \frac{2}{C^2} + \frac{\varepsilon^2(C\varepsilon+3)}{3(C\varepsilon+1)}. \quad (12)$$

The conditional probability distribution of $f(\boldsymbol{x})$ given $\boldsymbol{\Gamma}$ can instead be written as

$$P[f(\boldsymbol{x})|\boldsymbol{\Gamma}] = \frac{1}{\sqrt{2\pi}\sigma_t}\exp\{-\frac{(f(\boldsymbol{x})-f^*(\boldsymbol{x}))^2}{2\sigma_t^2}\}, \quad (13)$$

with

$\sigma_t^2(\boldsymbol{x}) = K(\boldsymbol{x},\boldsymbol{x}) - K_{\boldsymbol{X}_M,\boldsymbol{x}}^T K_{\boldsymbol{X}_M,\boldsymbol{X}_M}^{-1} K_{\boldsymbol{X}_M,\boldsymbol{x}}$ and $\boldsymbol{X}_M$ is the set of all the SVs.

Consequently, the error bar width of the prediction corresponding to the test input point $\boldsymbol{x}$ is

$$\sigma^2(\boldsymbol{x}) = \sigma_\delta^2 + \sigma_t^2(\boldsymbol{x}) = \frac{2}{C^2} + \frac{\varepsilon^2(C\varepsilon+3)}{3(C\varepsilon+1)} + K(\boldsymbol{x},\boldsymbol{x}) - K_{\boldsymbol{X}_M,\boldsymbol{x}}^T K_{\boldsymbol{X}_M,\boldsymbol{X}_M}^{-1} K_{\boldsymbol{X}_M,\boldsymbol{x}}. \quad (14)$$

3.1.3 Tuning of hyperparameters

The challenge for applying PSVR is to find the best hyperparameters values, as they play a critical role in the performance of the PSVR model.

A main advantage of PSVR is that it can provide an error bar estimation along with the predicted value. According to the special output of PSVR, a simple and effective strategy is proposed for the tuning of hyperparameters.

The strategy proposed to determine the best values for the three hyperparameters is a simple but effective grid search based on interpolation. Each parameter is initially selected within a given interval. The best values are to be found by minimizing the following criterion

$$C_1 \sum_{i=1}^N \sigma_i + C_2 \sum_{i=1}^N |\hat{y}_i - y_i| \quad (15)$$

where σ_i is the error bar width, $\hat{y}_i = f(\boldsymbol{x}_i)$ and y_i are separately the predicted value and the target value of the i^{th} input data point. C_1 and C_2 are the two weights of the two parts of the objective function (Equation (15)), the error bar width and the bias of the prediction. If C_1 is smaller than C_2, it means that we pay more attention to the variance of the prediction (error bar width) than to the accuracy in the prediction (distance between target and predicted values), and vice versa for C_1 bigger than C_2.

Compared to the strategy of finding the best hyperparameters values by the minimization of the prediction errors, the proposed strategy avoids the case where the prediction accuracy is very high but the prediction interval are very large. The grid search method used in tuning hyperparameters is efficient and uses less time than GA, PSO, etc. Normally the GA and PSO can find better values for hyperparameters but compared to the improvement on the prediction accuracy, the time consumed is not necessary. The grid search method can already give good enough results.

3.1.4 Application in the reference case study

One-day ahead prediction with PSVR for Scenario 1 in Table II is carried out as the case study. Partial autocorrelation calculates the correlation between the target and different time laps and three historical values are chosen as the inputs according to Figure 7. We fix $C \in [10, 10^5]$, $\gamma \in [10^{-7}, 10^3]$, $\varepsilon \in [10^{-3}, 10^{-1}]$, $C_1 = 4$ and $C_2 = 5$ by a trial-and-error process. For each parameter, a geometric sequence included in the corresponding interval is considered. In this applicative context, geometric sequences are better than arithmetic ones, since the parameter's influence on the objective function (Equation (10)) is highly non-linear. For C, ε and γ, geometric sequences of size 4, 10 and 4 are formed respectively. Note that for different training data sets, the best values of the parameters can change: hence, the tuning of the parameters in a feasible computational time is a relevant issue. In this case, the optimization of the objective function (Equation (9)) leads to the following choice for C, ε and γ: (6309.6, 0.0032, 7).

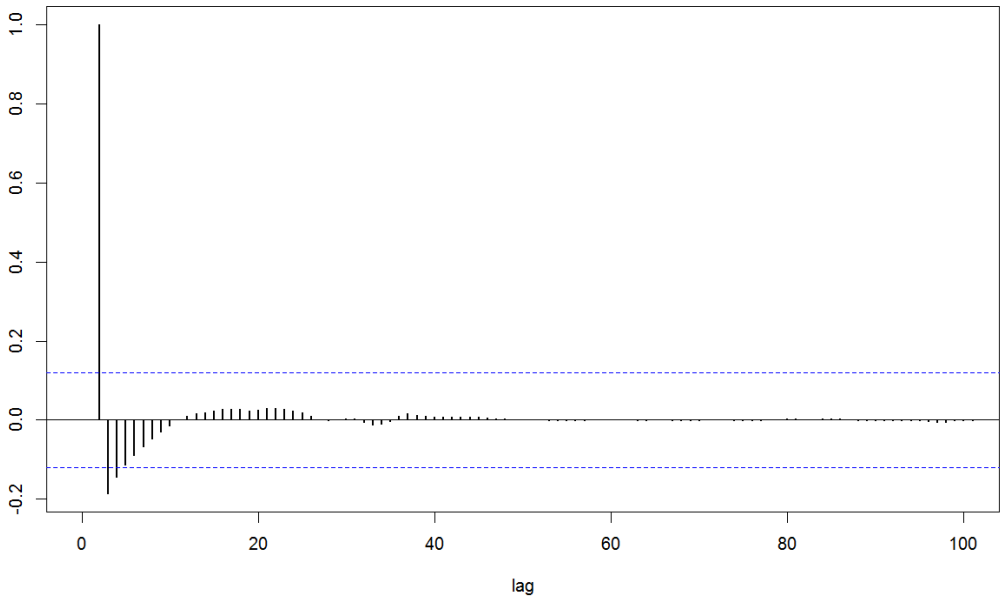

Fig. 7 Empirical partial autocorrelation function of the time series of the target values with respect to time laps.

The prediction interval empirical coverage estimated on the whole test dataset is 91.50%. The MSE is $5.4332*10^{-5}$. The Mean Relative Error (MRE) is smaller than 4%. If the model is trained using a bigger training data set, the relative error and absolute error will decrease. Comparison with Auto-Associative Kernel Regression (AAKR) method proves the accuracy of PSVR on this case study.

3.2 Training a SVR model on Feature Vectors (FVs)

3.2.1 Feature Vector Selection (FVS)

In [5], the authors propose a FVS method to select a subset of the training data points (i.e. FVs), which can represent the dimension of the whole dataset in RKHS. The other data points can all be expressed as a linear combination of the selected FVs in RKHS.

Suppose $(\boldsymbol{x}_i, y_i)$, for $i = 1, 2, \dots, N$ are the training data points and the mapping $\varphi(\boldsymbol{x})$ maps each input vector $\boldsymbol{x}_i$ into RKHS with the mapping $\boldsymbol{\varphi}_i$, for $i = 1, 2, \dots, T$. The kernel $k_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the inner product between $\boldsymbol{\varphi}_i$ and $\boldsymbol{\varphi}_j$. Suppose that the FVs selected from the training dataset are $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_L\}$ and the corresponding mapping is $S = \{\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2, \dots, \boldsymbol{\varphi}_L\}$: the process for selecting the new next FV is to calculate $\{a_{new,1}, a_{new,2}, \dots, a_{new,L}\}$ which gives the minimum of Equation (16), with $\boldsymbol{\varphi}_{new}$ being the mapping of the new input vector $\boldsymbol{x}_{new}$:

$$\delta_{new} = \frac{\left\| \boldsymbol{\varphi}_{new} - \sum_{i=1}^{L} a_{new,i} \boldsymbol{\varphi}_i \right\|^2}{\|\boldsymbol{\varphi}_{new}\|^2}. \tag{16}$$

The minimum of δ_{new} can be expressed with an inner product, as shown in Equation (17):

$$\min \delta_{new} = 1 - \frac{K_{S,new}^t K_{S,S}^{-1} K_{S,new}}{k_{new,new}}, \tag{17}$$

where $K_{S,S} = \left(k_{i,j}\right), i, j = 1,2, \dots, L$ is the kernel matrix of S and $K_{S,new} = \left(k_{i,new}\right), i = 1,2, \dots, L$ is the vector of the inner product between $\boldsymbol{\varphi}_{new}$. The expression

$$J_{S,new} = \frac{K_{S,new}^t K_{S,S}^{-1} K_{S,new}}{k_{new,new}} \tag{18}$$

is the local fitness of $\boldsymbol{x}_{new}$ with respect to the present feature space S. If $1 - J_{S,new}$ is zero, the new data point is not a new FV; otherwise, it is a new FV and is added to S. The multipliers $\boldsymbol{a}_{new} = \{a_{new,1}, a_{new,2}, \dots, a_{new,L}\}$ can be calculated by

$$\boldsymbol{a}_{new} = K_{S,new}^t K_{S,S}^{-1}. \tag{19}$$

With the global fitness defined as in Equation (20), the FVS procedure proceeds to select a subset of training data points with minimal size, which gives zero global fitness. The details for FVS is shown in Figure 8.

$$J_S = \sum_{i=1}^{T} J_{S,i} \tag{20}$$

In order to be more effective, a positive threshold ρ is introduced to make the FVS procedure faster. Different from the original work, after the selection of a new FV, the training dataset is reduced as $\boldsymbol{T}_r = \boldsymbol{T}_r \backslash \mathbf{E}$ with $\mathbf{E} = \{(\boldsymbol{x}_k, y_k)\ and\ (\boldsymbol{x}_i, y_i): 1 - J_{\boldsymbol{S},i} \leq \rho\}$. This can faster the selection process.

Initialization:
Training dataset: $\boldsymbol{T}_r = \{(\boldsymbol{x}_i, y_i)\}$, for $i = 1, 2, \ldots, T$
Feature space: **S** = [ ]
Threshold of local fitness: ρ
FVS:
First FV in **S**:
For i = 1 to T, calculate
$\mathbf{S} = \{\boldsymbol{x}_i\}$, compute the global fitness J_S for all training data points with respect to the present **S**.
End for.
Select the point which gives the maximum global fitness as the first FV and add it to **S** as the first FV.
$\boldsymbol{T}_r$ is reduced to the complement of **S** in $\boldsymbol{T}_r$, i.e. $\boldsymbol{T}_r = \boldsymbol{T}_r \backslash \mathbf{S}$.
Second and the other FVs:
Calculate the local fitness for all data points in $\boldsymbol{T}_r$ with respect to the present feature space **S**.
Select the data point k which gives the maximum of the local fitness:
If $1 - J_{S,k} > \rho$, this point is a new FV and is added to **S**; and $\boldsymbol{T}_r = \boldsymbol{T}_r \backslash \mathbf{E}$, with $\mathbf{E} = \{(\boldsymbol{x}_i, y_i) : 1 - J_{S,i} \leq \rho$ and $(\boldsymbol{x}_k, y_k)\}$;
If $1 - J_{S,k} \leq \rho$, end the process of FVS.

Fig. 8 Pseudo-code for FVS.

3.2.2 Train a SVR Model on FVs

When training a SVR model on the selected FVs, in order to keep the generalization ability and avoid the overfitting problem, the model is trained with respect to minimize the MSE on the whole training dataset, i.e. $\boldsymbol{\omega}$ in Equation (2) is a kernel expansion of the FVs, and the objective function in Equation (2) is still the minimization on the whole *N* training data points.

4. SVR-BASED ENSEMBLE FOR FAILURE PROGNOSTICS

For the situation of a large dataset without pattern drifts, the implementation of a single SVR model is time-consuming, as the computational complexity of training a SVR model increases exponentially with the size of the dataset. An ensemble approach is a better choice for such a situation. In this Section, different strategies for building SVR-based ensembles are introduced and details are reported in Papers II and III of Part IV. One novelty of the proposed ensembles is the dynamic-weighting strategy that calculates dynamically the weights of sub-models for each new input vector, before knowing the true output value, while most of the dynamic-weighted ensemble approaches in the literature update the weights according to prediction accuracy.

4.1 Basics of ensemble models

An ensemble-based approach is obtained by training diverse sub-models, and, then, combining their results with proper strategies. It can be proven that this can lead to superior performance with respect to a single model approach [7]. Ensemble-based approaches attempt to take advantage of each sub-model, by fusing results from all the sub-models. A simple paradigm of a typical ensemble-based approach is shown in Figure 9.

For building an ensemble, we need to answer three questions: how to divide the whole training dataset into different sub-dataset for each sub-model and to maximize the diversity between different sub-datasets; how to calculate the weights of each sub-model in such a way that the correct decisions are amplified, while the incorrect ones are counteracted.; how to combine the results from different sub-models.

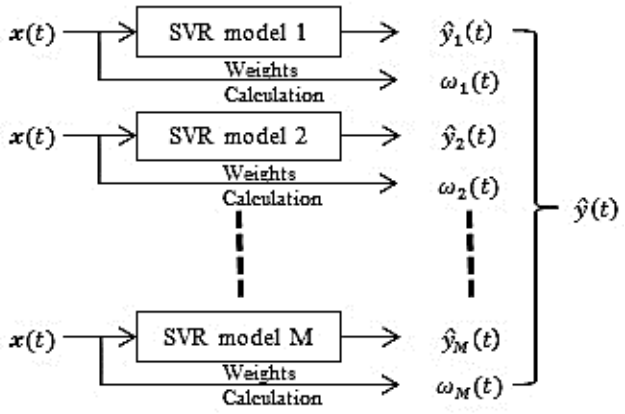

Fig. 9 Paradigm of an ensemble approach.

4.2 Strategies for deciding the sub-dataset for each sub-model

In the case of multiple scenarios, a natural strategy is to train a SVR model on each scenario. Then each sub-model is specified for a certain development of failure.

The second strategy is to divide the whole training dataset into different groups according to their output value, i.e. divide the data points into different groups where the output values of each group are in a certain interval. This method is efficient for time series data without abrupt changes where the inputs are some historical values of the output. The input values and output value are not so different. The sub-model trained on a group of data points is an expert on data points whose output values are in the range of outputs values of this group.

The third way is for the SVR model using Radial Basis kernel Function (RBF). The mapping related to RBF maps the data points into the high-dimensional RKHS where the norm of each data points is unit and the difference between different data points is only the angle. Thus, in the third strategy for deciding the training dataset for each sub-model, an angle-clustering algorithm, as shown in Figure 10, is used to divide all the training data points into a prefixed number (e.g. c) of clusters. Then, a SVR model can be trained on data points of each cluster.

For training dataset $\boldsymbol{T} = \{(\boldsymbol{x}_i, y_i)\}, i = 1,2, \dots, T$, choose the number of clusters c. Initialize a random cluster centers vector $(\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_c)$ chosen from the dataset $\boldsymbol{T}$.

Repeat: for $l = 1,2, \dots$

Compute the angle

$$D_{ik} = \arccos(k(\boldsymbol{x}_k, \boldsymbol{v}_i^{(l)})), 1 \le i \le c, \quad (7)$$

$1 \le k \le T$. $k(\cdot,\cdot)$ is the RBF.

Select the points of minimal distances for a cluster, and they belong to that cluster. Suppose $\{(\boldsymbol{x}_i, y_i)\}, i = 1,2, \dots, N_i$ are data points in cluster i,

Calculate the cluster centers

$$\boldsymbol{v}_i^{(l+1)*} = \frac{\sum_j^{N_i} \boldsymbol{x}_j}{N_i}.$$

Choose the nearest data point to the calculated cluster center in the corresponding cluster to be the new cluster center.

$$D_{ik}^* = \arccos\left(k\left(\boldsymbol{x}_k, \boldsymbol{v}_i^{(l+1)*}\right)\right), 1 \le k \le N_i \text{ and } \boldsymbol{v}_i^{(l+1)} = argmin_{\,i}(D_{ik}^*).$$

Until $|D^{(l+1)} - D^{(l)}| \le \tau, \tau$ is a small positive value and $D^{(l)} = \sum_{i=1}^{c} \sum_{k=1}^{T} D_{ik}$.

Fig. 10 Pseudo-code of angle-clustering algorithm.

In order to reduce the computational burden of training a SVR model with large dataset, FVS proposed in [5] is used to select a small part of the whole training data points to train the SVR model, whereas the other data points can all be represented by a linear combination of the selected data points (FVs). Then a SVR is trained on the FVs with strategy proposed in Section 3.2.2.

4.3 Combination of the outputs from different sub-models

An ensemble-based approach is obtained by combining diverse models, to obtain superior performance with respect to that obtained with a single model. The strategy of combining different algorithms into an ensemble has been found attractive in a wide variety of research fields. There are two ways for obtaining a final result from those of different sub-models: one is to select the best sub-model in the ensemble, and to use it to give the prognostics, while the other one is to obtain the prediction as weighted sum of the sub-model results [28]. The latter one is most used in recent research.

In this thesis, all ensemble approaches integrate a dynamic-weighting strategy. In the literature on ensemble methods, weights of sub-models are calculated during the training part by maximizing the prediction accuracy, and few of them change the weights during the prediction, where the weights are modified only when one or several sub-models are updated with new data points, or new sub-models are added to the ensemble. The dynamic-weighting strategy proposed in this thesis is to calculate the weight of each sub-model for each data point, i.e. for different data points the weights of sub-model are recalculated. The reason behind this is that sometimes the ensemble cannot give good prediction results because the sub-models giving good results are not given more important weights. During the prediction, one sub-model performs well only on certain data points considering its training data points, and bad on the others. Thus, dynamic-weighting strategy needs to be integrate in ensemble approach to adapt the weights to different data points.

The output of the ensemble is a weighted sum of the outputs of all the outputs of sub-models, as shown in Equation (21). In the equation, the weights are a function of time t and depend on the data points.

$$\hat{y}(t) = \sum_{j=1}^{M} \omega_j(t)\hat{y}_j(t) \quad (21)$$

4.4 Calculating the weights for different sub-models

Two strategies are proposed to calculate the weights of sub-models for each data point. One is based on the fuzzy similarity between the new data point and the training data points as proposed in [179]; the other one is based on local fitness calculated by Equation (18). The proposed approach can calculate dynamically the weights depending on different data points and the weights are assigned before knowing the true output of the data point.

4.4.1 Weights calculation based on fuzzy similarity analysis

Suppose there are totally N_i training data points for the *i*-th sub-model and the new data point is $(\boldsymbol{x}(t), y(t))$. First we calculate the Euclidean distance between the new data point and all the training data points of the *i*-th sub-model and find the minimal Euclidean distance, suppose it is $d_i(t_0)$. The second step is use Equations (22) and (23) to calculate the raw weight of the *i*-th sub-model for the new data point, as proposed in [179]. Suppose there are M sub-models, and the third step is to normalize the weights for all sub-models with Equation (24).

$$\mu = \exp(-(-\ln(\alpha)/\beta^2) d_i(t_0)^2) \tag{22}$$

$$w_i = \mu \exp(-\frac{1-\mu}{\beta}) \tag{23}$$

$$w_i = w_i / \sum_{j=1}^{M} w_j \tag{24}$$

In Equation (22), the arbitrary parameters α and β can be set by the analyst to shape the desired interpretation of similarity into the fuzzy set: the larger the value of the ratio -ln(α) / β^2, the narrower the fuzzy set and the stronger the definition of similarity.

4.4.2 Weights calculation based on local fitness

For a new data point, first we calculate the local fitness $J_{i,new}$ of this point with respect to the selected FVs from the training data points of the *i*-th sub-model with Equation (18). Then Equation (25) is used to calculate the weights of each sub-model for this new data point.

$$\omega_i = \frac{1/(1-J_{i,new}+\tau)}{\sum_{j=1}^{M} 1/(1-J_{j,new}+\tau)} \tag{25}$$

In Equation (25), τ is a very small value so that it works in the case $J_{i,new} = 1$.

Note that the weights calculated in Sections 5.3.1 and 5.3.2 are all functions of the new data point, i.e. they change with different data points.

4.5 Applications in reference case study

In the case study, we consider all the scenarios in Table II available for training SVR-based ensemble with the methods proposed for building different sub-models, calculating the sub-models' weights and fusing the prediction results from sub-models.

Three strategies for building an ensemble are proposed in the related papers, noted Ensemble 1, 2 and 3. Ensembles 1 and 2 train sub-models with the first and second strategies proposed in Section 4.2 and the weights are calculated with fuzzy similarity analysis presented in Section 4.4.1. Ensemble 3 integrate the third strategy in Section 4.2 for training diverse sub-models and their weights are calculated with local fitness in FVS presented in Section 4.4.2. For comparisons, a single SVR model and a fixed-weighted ensemble are applied for the case study as benchmark approaches. Figure 11 shows the boxplot of the MRE on different scenarios given by the proposed ensembles and the benchmark approaches. It is clear that the proposed ensembles give better results than the benchmarks. We can also note that the proposed ensembles give different prediction results as different strategies are used in the ensemble. Ensemble 3 gives the worst result as the FVS selects only a small part of the training dataset to train the model, and it is inevitable that some useful information are neglected by the two tolerance parameters as shown in Section 3.2.1. Ensemble 2 gives worse result than Ensemble 1 because the training dataset for some sub-models are not large enough.

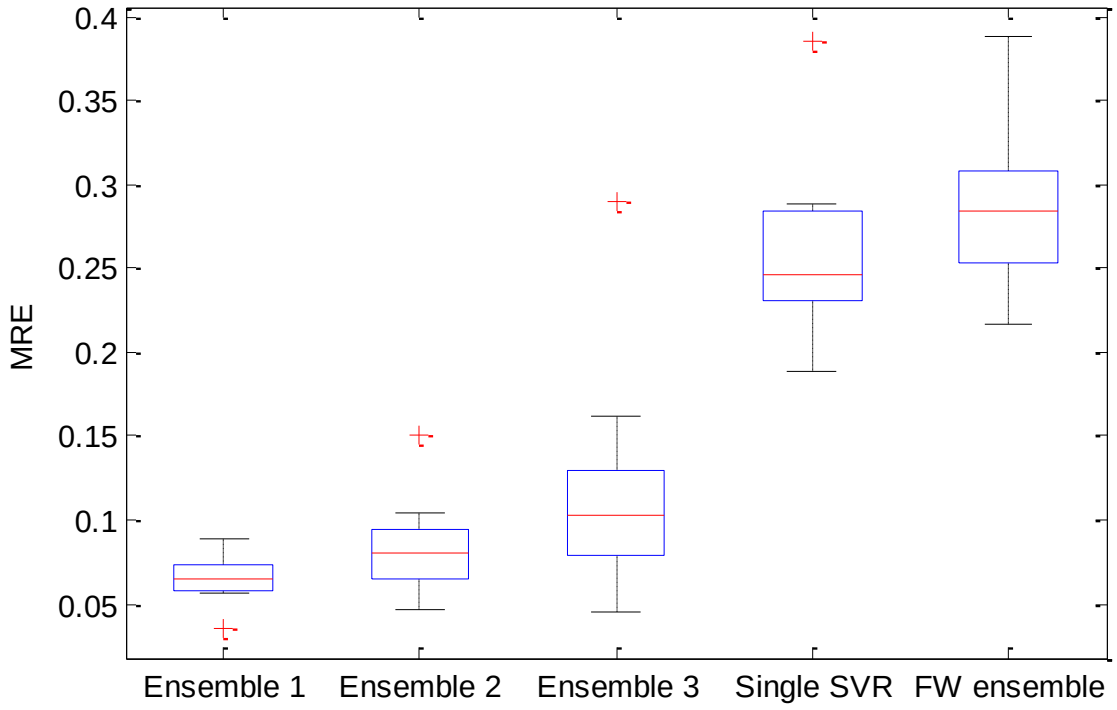

Fig. 11 Boxplot of the MRE on different scenarios.

5. ADAPTIVE LEARNING OF SINGLE SVR MODEL FOR FAILURE PROGNOSTICS

In the previous two Chapters, the training and test datasets are supposed to be generated in stationary environment, i.e. the data points follow an identical and independent distribution. In the case that the component is operated in nonstationary environment, the distribution generating the data points may change with time. In such case, we need to provide the model with adaptive learning ability. In this Chapter, we introduce the approach (named Online-SVR-FID) proposed in this thesis for adaptive online learning of single SVR model based on FVS. Details on the experimental results are reported in Paper IV of Part IV. In the next section, an online learning ensemble for drifting time series datasets is proposed.

5.1 Methodology

Two types pattern drifts are defined in the thesis and the update strategies for a single SVR model according to these two types of pattern drifts are proposed. A new data point is a new pattern (or new FV) if the mapping of its input vector in RKHS cannot be represented by a linear combination of the mapping of existing patterns, while it is a changed pattern if the mapping can be represented by such a linear combination but the bias of the 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 carefully 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.

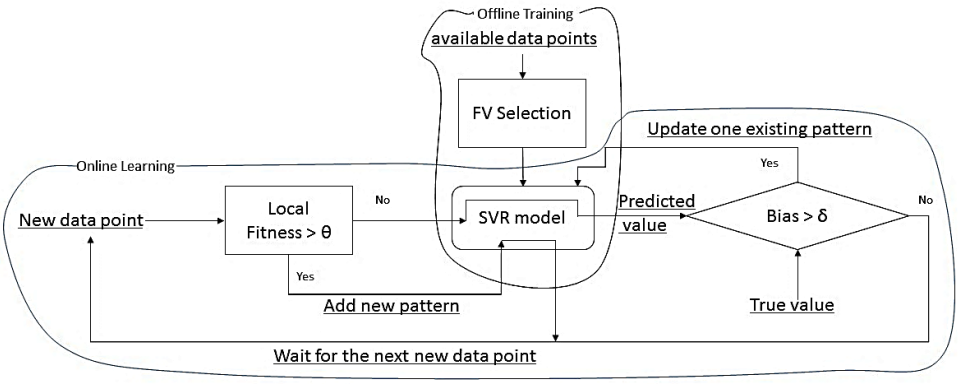

Fig. 12 Paradigm of Online-SVR-FID.

Figure 12 is the paradigm of Online-SVR-FID, and Figure 13 presents the pseudo-code of the proposed approach.

Initialization:

Training dataset: $\boldsymbol{T}_r = \{(\boldsymbol{x}_i, y_i)\}$, for $i = 1, 2, \ldots, T$

Testing dataset: $\boldsymbol{T}_e = \{(\boldsymbol{x}_i, y_i)\}$, for $i = T+1, T+2, \ldots, T+H$

Feature space: **S** = [ ]

Threshold of local fitness: ρ

Threshold of bias: δ

Offline Training:

First FV in **S**:

For *i* = 1 to T 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 **S** as the first FV.

$\boldsymbol{T}_r$ is reduced as the complement of **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 **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 **S**; $\mathbf{E} = \{(\boldsymbol{x}_k, y_k)\ and\ (\boldsymbol{x}_i, y_i): 1 - J_{\boldsymbol{S},i} \leq \rho\}$ and $\boldsymbol{T}_r$ is reduced as the complement of **E** in $\boldsymbol{T}_r$, i.e. $\boldsymbol{T}_r = \boldsymbol{T}_r \backslash \mathbf{E}$;

If $1 - J_{\boldsymbol{S},k} \leq \rho$, end the process of FVs selection;

Train the SVR model on the FVs in **S**.

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

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

If the bias is smaller than δ

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. (4). Unlearn this FV found with decremental learning 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.

Fig. 13 Pseudo-code of Online-SVR-FID.

When a new data point is judged as a changed pattern, it is used to replace one data point in the

present model. The process is as follow:

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 Equation (5) 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 $\boldsymbol{a}_N$ in Equation (16) for the new data point with Equation (19) according to the FVS introduced in Section 4.2. 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 [15] 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 [15] 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 in Equation (16) 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 Equation (17) 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.

5.2 Application in reference case study

In this section, Scenario 1 in Table II is selected as the data for the case study. The model is trained on the stable part of the data and then the anomaly data points are fed to the model simulating the online learning process. Comparisons are carried out with several popular online learning approaches for kernel methods, i.e. Incremental Learning in [15], Naïve Online R_{reg} Minimization Algorithm (NORMA) in [67], SOGP in [16] and Kernel-based Recursive Least Square Tracker (KRLS-T) in [79]. 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 in the case study.

Table III Comparisons of online prediction results with Online-SVR-FID, Incremental Learning, NORMA and SOGP.

	Online-SVR-FID	Incremental Learning	NORMA	SOGP	KRLS
MSE	0.0011	0.0013	1.7091	0.0019	0.0044
MRE	0.0561	0.0548	2.9965	0.0763	0.0779
NMSE	0.0056	0.0069	8.854	0.0098	0.0228
Online Learning time (s)	9.2067	1354.6425	3191.8332	332.7395	9.5970
Model size before Online Learning	11	300	300	25	200
Model size after Online Learning	14	800	269	60	200

6. ADAPTIVE LEARNING OF SVR-BASED ENSEMBLE FOR FAILURE PROGNOSTICS

A main disadvantage with Online-SVR-FID is that some past patterns may be deleted during the UPDATE process in Figure 13. When the deleted patterns reoccur, the updated SVR model needs to relearn them from scratch, which increase the computational burden and reduces the prediction accuracy these patterns. In the case of a large dataset with pattern drifts, applying Online-SVR-FID on single SVR model is also time-consuming. In this Chapter, an online learning ensemble approach is proposed based on FVS to store all the past patterns. The proposed approach is named OE-FV. OE-FV builds automatically an ensemble from a single SVR 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 reoccur, the most relevant sub-models are selected to derive the prediction. Details on the experimental results are reported in Paper V of Part IV.

6.1 Methodology

The main procedure is shown in Figure 14. OE-FV builds an ensemble sequentially from the first model, named M_1 that is trained on the preliminary training dataset. All the other sub-models can be seen as a "copy" of M_1 at one instance during the developing process. These sub-models are expected to be different from each other and represent the data at a certain period. Only the sub-model M_1 is adaptively updated with new data points, while the other sub-models are fixed once created.

1. Train a model M_1 with kernel methods on the training dataset.
2. Suppose there are *n* sub-models ($M_1, M_2, \ldots, M_n$) in the ensemble when a new data point is coming:
   - 2.1 Calculate the predicted value for the new data 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 the model 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;

       2.2.1.1 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 M_1 is up-to-date;

       2.2.1.2 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;

   - 2.4 Update the prediction error **Er** of each sub-model.

Fig. 14 The main procedure of OE-FV.

6.1.1 Training of the first sub-model in Ensemble

A single model M_1 is trained on the training dataset which is also the first and basic sub-model in the ensemble (step 1 in Figure 14). In order to reduce the model complexity and computational burden, 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 keeps 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.

6.1.2 Calculation of the predicted value of a new data point

When a new data point is coming, in order to give a reasonable prediction using the ensemble approach (step 2.1 in Figure 14), we use a dynamic ensemble selection strategy. A dynamic ensemble selection is to select the sub-models that are most relevant to the new data point to calculate their separate prediction, and, 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. In OE-FV, they are selected by the local fitness of the new data point, calculated by Equation (18), with respect to the FVs in each sub-model. 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 (26). 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 (27), with $\hat{y}_i$ and $\hat{y}$ separately the predicted value of selected sub-models and the ensemble.

$$\boldsymbol{\omega} = \frac{1/\boldsymbol{Er}_{Eoc}{}^2}{\sum 1/\boldsymbol{Er}_{Eoc}{}^2} \tag{26}$$

$$\hat{y} = \sum_{EoC} \omega_i \hat{y}_i \tag{27}$$

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, then, used for calculating the prediction of the ensemble. In Equations (26), $\mathbf{Er}_{Eoc}$ is replaced by $\mathbf{Er}$ and in Equation (27), the weighed sum is carried out on all the sub-models.

6.1.3 Update of the ensemble with a new pattern

If the local fitness of the new data point with respect to the FVs in each sub-model satisfies the relation $1 - J_{Si}(\boldsymbol{x}) > \rho$, it is judged as a new FV, and it is added to the first model M_1 that is trained on the training dataset (step 2.2 in Figure 14). The other sub-models are not modified with the new FV, as they represent only the patterns in the data at 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 is increased by one, the model is retrained through minimizing the MSE on the recent data points (How to choose the recent data points is explained in details in Section 6.1.6).

6.1.4 Update of the ensemble with a changed pattern

Once the new data point is judged as not a new FV, the verification of a changed FV is carried out by calculating the prediction errors (absolute bias between the predicted value and the true output) of all the sub-models. If the prediction errors are all bigger than the preset threshold δ, the new data point is judged as a changed pattern. It is used to replace a 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 an idea for SVR which counts the times of being a support vector in the past SVR models during the adaptive learning process, and 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 its contribution through a weighted sum of its value calculated in Equation (19) 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 (19) can give its similarity with each FV in M_1. A bigger a_i in $\boldsymbol{a}$ represents a larger

similarity, thus, a bigger contribution to the prediction of the new data point. Its contribution is 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 2.2.1.1 in Figure 14 proposes to "copy" the model M_1 as a new sub-model and before the replacement, then, 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 represent different steps of the data stream.

If the selected FV in M_1 is not a unique in the ensemble, it is replaced directly by the new data point without adding a new sub-model (step 2.2.1.2 in Figure 14).

6.1.5 Update of the prediction error of sub-models

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 in Figure 14, 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 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 are 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 $\boldsymbol{Er}_{Eoc}$ their prediction errors, β a positive parameter smaller than one and $\hat{\boldsymbol{y}}_i$ is the 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) For M_1, it is different from the above two types of the 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 in Figure 14, 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 in Figure 14. M_1 gives a new prediction for the new data point different from the one calculated in step 2.1 in Figure 14 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 according to $Er_1 = \beta Er_1 + |\hat{y}_{1,new} - y_i|$, with $\hat{y}_{1,new}$ is the prediction for the new data point given by updated M_1.

4) If a new sub-model is created during the online learning of the new data point, the prediction error the new sub-model is calculated with $Er_{n+1} = \beta Er_1 + |\hat{y}_{1,old} - y_i|$, with $\hat{y}_{1,old}$ the prediction for the new data point given by M_1 at step 2.1 in Figure 14 which is not updated yet with the new data point, and Er_1 is the prediction error of M_1 at the beginning of this iteration in step 2.1 in Figure 14, i.e. before updating.

6.1.6 Retraining of the sub-model M_1

Facing a new FV or a changed FV, the model M_1 needs to be updated. However, it is not always possible to find a way to update 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.

Training a classic kennel-based model takes the minimization of the MSE on the training dataset as the objective function. In this paper, M_1 is trained on the FVs and minimizes 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 number of data points considered in the objective function is to minimize the MSE on the data points from i_0 to i. In order to avoid the

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 is fixed during the retraining of M_1, i.e. the number of the recent data points for retraining M_1 is $\min(\max(N_{min}, i - i_0), N\_max)$.

6.1.7 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, OE-FV can timely learn the new patterns compared to data chunk-based and drift detector-based approaches for online learning ensemble. It can instantly follow the pattern drift in the data, and the online learning ensembles based on data chunk or sliding window can only react after a sufficient number of new data points is available.

The aim of 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.

When a new sub-model need to be created, there is no need to train this new sub-model, as it is a "copy" of the 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 between 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.

The new data points are all used to update the sub-models' weights, and only few of them are used to update the M_1 and create new sub-models. For each new data points, 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 strategies can reduce the computational complexity of the online learning process.

6.2 Application in reference case study

OE-FV is applied to the case of drifting data stream. The first model M_1 is trained on the first 500 data points in the first scenario in Table II, and the other data point in scenario 1 and all the data points of the other scenarios are fed to the ensemble simulating the drifting data stream.

Comparisons of experimental results are carried out among Online-SVR-FID, Learn++.NSE,

OWE and the proposed OE-FV, considering the prediction accuracy and the computation time. 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 introduced in Section 5. In the experiment, there are totally 1198 new data points judged as changed patterns and 13 new data points as new patterns. While 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, thus the computational complexity, as all the patterns are stored in the ensemble. Thus, OE-FV solves the problem of Online-SVR-FID with recurring patterns.

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, while Learn++.NSE gives the worst and OE-FV gives 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 that 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 results than data chunk-based or sliding window-based ones in frequently changing environment.

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-model 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 sliding window.

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 reoccur, it has to relearn them before giving a good prediction result. This disadvantage increases the number of updating actions during online learning, thus the computational burden, and decreases the prediction accuracy. While 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 irrelative 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. A larger maximal number of sub-models doesn't always increases the accuracy. The accuracy is no longer improved when the number of sub-models is bigger than a certain value.

7. A NOVEL GEOMETRICALLY INTERPRETABLE KERNEL-BASED MODEL TRAINED WITH FVS: REDUCED RANK KERNEL RIDGE REGRESSION-II (RRKRR-II)

The main drawbacks of standard kernel methods are the unacceptable computational burden for training with large infinite datasets, the difficulty in tuning the hyperparameters and the lack of interpretability of the model. by analyzing the distribution property of the inner product (the kernel function is an inner product of two vectors in RKHS) and the geometrical relation between a training data point and the FVs selected with FVS, a geometrically interpretable approach is proposed for regression and prediction, which describes the linear relation between the predicted values of FVs and that of any data point under static environment. FVS is used to select the FVs which can represent the dimensions of the training dataset in RKHS, and the linear relation between the predicted value of the FVs and those of the other data points are derived from the general form of the estimate function for kernel methods of Equation (1). In order to keep all the information contained in the selected FVs, an optimization problem with equal constraints (similar to a Least Square-Support Vector Machine (LS-SVM)) is formed to find the minimal MSE (without regularization term in Equation (2)) on the whole training dataset (not only on the selected FVs). Thus, the unknowns in the estimate function of the proposed approach are the predicted values of the FVs and a constant (zero or nonzero), which can be calculated analytically. Minimizing the MSE on the whole training dataset of the model built on the selected FVs can guarantee the generalization performance of the model, even without a regularization term. Equal constraints in the optimization problem keep all the information in the FVs (i.e. no FV is ignored through the loss function, as in SVR) and the optimal values for the unknowns can be calculated analytically. Experimental results are detailed in Paper VI of Part IV.

7.1 Methodology

Suppose $\boldsymbol{S} = (\boldsymbol{x}_i, y_i), i = 1,2, \dots, M$ are the FVs selected with FVS from the training dataset $\boldsymbol{T}$; for any data point $(\boldsymbol{x}, y)$, its mapping $\boldsymbol{\varphi}(\boldsymbol{x})$ in RKHS can be expressed as a linear combination of the mapping of the selected FVs, i.e. $\sum_{i=1}^{M} a_i \boldsymbol{\varphi}(\boldsymbol{x}_i)$ and $a_j, j = 1,2, \dots, M$ are multipliers calculated with Equation (19). Note that $f(\boldsymbol{x})$ in Equation (7) can also be written as $f(\boldsymbol{x}) = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \boldsymbol{\varphi}(\boldsymbol{x}) \rangle + b$, and it can be rewritten as:

$$f(\boldsymbol{x}) = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)\langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \sum_{j=1}^{M}\beta_j \boldsymbol{\varphi}(\boldsymbol{x}_j)\rangle + b. \quad (28)$$

By the mathematical distribution property of the inner product, Equation (28) equals to

$$f(\boldsymbol{x}) = \sum_{i=1}^{N}\sum_{j=1}^{M} a_j\,(\alpha_i - \alpha_i^*)\langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \boldsymbol{\varphi}(\boldsymbol{x}_j)\rangle + b$$

$$= \sum_{j=1}^{M} a_j(\sum_{i=1}^{N}(\alpha_i - \alpha_i^*)\,\langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \boldsymbol{\varphi}(\boldsymbol{x}_j)\rangle) + b. \quad (29)$$

Equation (29) can be further written as

$$f(\boldsymbol{x}) = \sum_{j=1}^{M} a_j(f(\boldsymbol{x}_j) - b) + b, \quad (30)$$

where $f(\boldsymbol{x}_j), j = 1,2,\dots,M$ are the predicted values of the FVs in $\boldsymbol{S}$, b is a constant variable and $a_j, j = 1,2,\dots,M$ are the multipliers calculated with Equation (19).

Now, the new form of the estimate function of Equation (7) can be written as

$$g(\boldsymbol{x}) = \sum_{j=1}^{M} a_j(\boldsymbol{x})(\hat{y}_j - b) + b, \quad (31)$$

where $\hat{y}_j, j = 1,2,\dots,M$ are the predicted values of the FVs selected from the training dataset $\boldsymbol{T}$, b is a constant value, $g(\boldsymbol{x})$ is the prediction of any data point $(\boldsymbol{x}, y)$ and $a_j(\boldsymbol{x})$ is the j-th value of the vector $\boldsymbol{a}(\boldsymbol{x})$ calculated with Equation (19), which is dependent only on the input $\boldsymbol{x}$ once the FVs are selected.

Equation (31) describes the linear relation between the predicted values of FVs, i.e. $\hat{y}_j$ and that of any other data point, i.e. $g(\boldsymbol{x})$. This new prediction model is called RRKRR-II. Equation (31) shows that if we know the predicted values for the FVs and the constant b, we can give directly the predicted value for any data point. In the next sub-section, the analytic solutions for the unknowns in Equation (31), i.e. $\hat{y}_j, j = 1,2,\dots,M$ and b are given.

The optimization problem for RRKRR-II is defined as

$$minimize_{\hat{y}_j,b} \qquad W = \frac{1}{N}\sum_{i=1}^{N}(g(\boldsymbol{x_i}) - y_i)^2$$

$$subject\ to \qquad g(\boldsymbol{x_i}) = \sum_{j=1}^{M} a_j(\boldsymbol{x_i})(\hat{y}_j - b) + b, \quad (32)$$

with M representing the number of FVs selected from the whole training dataset $\boldsymbol{T} = (\boldsymbol{x}_i, y_i), i = 1,2,\dots,N$. The optimization problem is trying to find the minimal MSE on the whole training dataset $\boldsymbol{T}$.

After replacing $g(\boldsymbol{x_i})$ in the objective function in Equation (30) with $\sum_{j=1}^{M} a_j(\boldsymbol{x_i})(\hat{y}_j - b) + b$, the objective function becomes

$$W = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=1}^{M} a_j(\boldsymbol{x_i})\hat{y}_j + b(1-\sum_{j=1}^{M} a_j(\boldsymbol{x_i})) - y_i\right)^2. \quad (33)$$

Setting the partial derivatives of W with respect to $\hat{y}_j$ and b to zero yields:

$$\frac{\partial W}{\partial \hat{y}_{j_0}} = \sum_{j=1}^{M}\sum_{i=1}^{N} a_{j_0}(\boldsymbol{x_i}) * a_j(\boldsymbol{x_i}) * \hat{y}_j + b * \sum_{i=1}^{N} a_{j_0}(\boldsymbol{x_i}) * \left(1-\sum_{j=1}^{M} a_j(\boldsymbol{x_i})\right) - \sum_{i=1}^{N} a_{j_0}(\boldsymbol{x_i}) * y_i = 0 \quad (34)$$

$$\frac{\partial W}{\partial b} = \sum_{j=1}^{M}\sum_{i=1}^{N} a_j(\boldsymbol{x}_i) * (1-\sum_{l=1}^{M} a_l(\boldsymbol{x}_i)) * \hat{y}_j + b * \sum_{i=1}^{N}\left(1-\sum_{j=1}^{M} a_j(\boldsymbol{x_i})\right)^2 - \sum_{i=1}^{N}\left(1-\sum_{j=1}^{M} a_j(\boldsymbol{x_i})\right) * y_i = 0. \quad (35)$$

These previous Equations (34) and (35) can be expressed as a system of equations as

$$\begin{bmatrix} \boldsymbol{\Omega} & \mathbf{H} \\ \boldsymbol{\Gamma}^T & c \end{bmatrix}\begin{bmatrix} \hat{\boldsymbol{y}} \\ b \end{bmatrix} = \begin{bmatrix} \mathbf{P} \\ l \end{bmatrix}, \quad (36)$$

where $\boldsymbol{\Omega}$ is a $M \times M$ matrix with $\boldsymbol{\Omega}_{mn} = \sum_{i=1}^{N} a_m(\boldsymbol{x_i}) * a_n(\boldsymbol{x_i})$, $\mathbf{H}$ is a $M \times 1$ vector with $\mathbf{H}_m = \sum_{i=1}^{N} a_m(\boldsymbol{x_i}) * \left(1-\sum_{j=1}^{M} a_j(\boldsymbol{x_i})\right)$, $\boldsymbol{\Gamma}$ is a $M \times 1$ vector with $\boldsymbol{\Gamma}_m = \sum_{i=1}^{N} a_m(\boldsymbol{x}_i) * (1-\sum_{l=1}^{M} a_l(\boldsymbol{x}_i))$, c is a constant and $c = \sum_{i=1}^{N}\left(1-\sum_{j=1}^{M} a_j(\boldsymbol{x}_i)\right)^2$; $\hat{\boldsymbol{y}} = (\hat{y}_j), j = 1,2,\dots,M$ and b are the unknowns in Equation (29), $\mathbf{P}$ is a $M \times 1$ vector with $\mathbf{P}_m = \sum_{i=1}^{N} a_m(\boldsymbol{x_i}) * y_i$, $l = \sum_{i=1}^{N}\left(1-\sum_{j=1}^{M} a_j(\boldsymbol{x}_i)\right) * y_i$.

The values of the unknowns in Equation (31) can be directly calculated by solving Equation (36).

The adaptive online learning approaches proposed in Sections 5 and 6 can all be applied on RRKRR-II.

7.2 Application in reference case study

The experiment is carried out on Scenario 1, 300 (at time instances 500 - 800) data points are selected as training dataset and the following 800 (at time instances 801 - 1600) data points forms the test dataset. The benchmark methods are Kernel Gaussian Process (KGP) [127], Kernel Partial Least Square (KPLS) [131], KRR []132], RRKRR [5], Relevance Vector Machine (RVM) [150] and SVR [141]. Table V shows the prediction results on the test dataset with different kernel regression methods considering time for training and testing, MSE and MRE.

We can see that RRKRR-II uses much less time than the other methods while giving comparable

results with the best results given by the benchmarks.

Table V Comparison of prediction results of the case study given by different methods

	RRKRR-II	KGP	KPLS	KRR	RRKRR	RVM	SVR
Time_train	**0.0558**	5.0651	0.0688	0.3059	0.2657	0.4296	0.2336
Time_test	0.0339	0.2205	0.0151	0.0053	0.0264	**0.0015**	0.0321
MSE	0.0086	0.1254	0.0062	**0.0013**	0.0264	0.0031	0.0287
MRE	0.1658	0.6824	0.1440	**0.0548**	0.3114	0.0554	0.3278

Experiment results on some public datasets can be found in Paper VI of Part IV.

8. CONCLUSIONS AND PERSPECTIVES

The aim of this Ph.D. research work is to develop SVR-based methods for failure prognostics, based on time series data. The object of analysis considered for the development of the models is a safety-critical component, the first seal in the RCP of a NPP: the importance of monitoring and prognostics for this component is extremely high. According to the amount of data and the underlying distribution generating the data, different approaches and strategies are proposed for tackling small/large datasets and stationary/nonstationary environments.

8.1 Conclusions and original contributions

Failure prognostics within maintenance engineering aims at predicting the future health of the SSC of interest on a short-term/long-term time horizon. The benefits of prognostic approaches include: warning of failures in advance; minimization of unscheduled maintenance; extended maintenance cycles; reduction of life-cycle cost of the SSC of interest by decreasing inspection cost and the SSC downtime; improved qualification of the SSC of interest, etc.

There are yet some challenges for prognostic approaches, including improving the robustness, adaptability and generalization power and estimating the uncertainty associated with the prediction. For the prognostic approach embraced in this thesis, i.e. SVR, there are also challenges for reducing the computational complexity and tuning the hyperparameters.

The original contributions of this thesis are summarized in Table VI, with respect to the challenges mentioned above for prognostic approaches and SVR.

Table VI Contributions of this thesis, with respect to the challenges for prognostic approaches and SVR.

	Contributions
Robustness	In Papers III, IV, and VI, the SVR models are trained on a small part of the training dataset, with the objective of minimizing the MSE on the whole training dataset.
Uncertainty quantification	In Papers I and II, PSVR is integrated to derive the error bar associated to the predicted value.
Adaptability	In Papers II and III, dynamic weighted ensembles are proposed to effectively modify the sub-models weights for each new input vector. In Papers IV and V, adaptive online learning approaches are proposed for single SVR and SVR-based ensemble models to manage new patterns and changed patterns in the new data points.
Generalization power	SVR-based approaches can be used for different SSC in different environments, not only for the NPP component considered in this thesis.
Reducing computational complexity	In Papers II and III, ensemble approaches are proposed to reduce the computational burden with large datasets. In Papers III, IV, V and VI, only part of the training dataset is elected to train the model. In Paper VI, the solution of RRKRR-II can be calculated analytically, instead of solving iteratively the dual problem.
Tuning hyperparameters	In Paper I, the grid search method is integrated for tuning the hyperparameters, which is proved to be faster than GA. In Paper VI, the hyperparameters are tuned with respect to the change of the MSE on the training dataset.

As the case study in this thesis considers different situations of the time series data available, i.e. small dataset without pattern drifts, large dataset without pattern drifts, small dataset with pattern drifts and large dataset with pattern drifts, different SVR-based approaches are proposed in this thesis to tackle the different situations.

For small datasets without pattern drifts, a single SVR model is trained with the proposed strategies for tuning hyperparamters. In order to reduce the computational complexity, FVS is integrated for reducing the size of the training dataset.

For large datasets without pattern drifts, training a single SVR becomes computationally burdensome, and strategies for building ensembles are proposed. Different approaches are proposed for building diverse sub-models and calculating their weights. The outputs of the sub-models are combined with a weighted-sum strategy. The main novelty of the proposed ensembles is dynamically calculating the sub-models weights for each test data point.

In the situations with pattern drifts, adaptive online learning approaches are proposed separately for single SVR model (Online-SVR-FID) and ensemble (OE-FV). Based on FVS, two types of pattern drifts are firstly defined: new patterns and changed patterns. Different actions are taken

to make sure that the single model/ensemble follow efficiently the current patterns. Online-SVR-FID can follow timely and precisely the ongoing patterns, but some past patterns are deleted from the model during the update process. OE-FV aims at solving this problem by storing all the past patterns in the ensemble. Each sub-model represents a certain period of the data. Dynamic ensemble selection is integrated in OE-FV to dynamically select the sub-models most relevant to the new data point to generate its predicted value. Dynamic ensemble selection before the prediction can reduce the influence of the irrelevant sub-models on the prediction results.

These previous approaches are all tested on the reference case study presented in Section 1.3. Comparisons with other approaches prove the efficiency and accuracy of the proposed approaches in the reference case study.

Considering the interpretability and the computational burden of a SVR model, a geometrically interpretable kernel method, i.e. RRKRR-II, is proposed based on FVS. RRKRR-II describes the linear relation between the predicted value of a new input vector and those of the FVs selected form the training dataset. The applications on five public datasets show the robustness and accuracy of RRKRR-II, compared to the popular kernel methods.

8.2 Future work

Various research directions can be taken to extend the work developed in this thesis. Important ones include the following perspectives:

- **The long-term prediction**, e.g. RUL prediction: different approaches can be integrated, e.g. combination of long-term and short-term predictions, combination of model-based and data-driven approaches, etc.
- **The application of prognostics for maintenance decision-making**, e.g. maintenance planning: the automatic and dynamic maintenance scheduling can reduce investment in hardware and personnel, and based on the prediction of the future health of the SSC of interest, the scheduled maintenance can be replaced by a dynamic one where the maintenance is carried out only if the risk of failure exceeds a certain threshold.
- **The uncertainty quantification of the prediction given by SVR**: in most of the work in this thesis, uncertainties in the prediction, which can be caused by measurement noise, model uncertainties and missing or unavailable training data, are not explicitly considered, while it is very important to know the confidence that can be put in the prediction results

for confident decision-making.

- **The propagation of uncertainty for decision making**: once the uncertainty of the prediction of SVR can be quantified, a real problem is how to calculate the influence it has on the decision making process.
- **The interpretability of SVR**: data-driven approaches are normally difficult to validate and verify before deployment, and how to reflect the case-specific physical characteristics is a main challenge for SVR. Although RRKRR-II tries to make the model easier to understand, it is far from enough.
- **A very interesting work considering RRKRR-II** is to calculate the uncertainty of the predicted values of selected FVs. If a distribution can be calculated for the predicted value of each selected FV, the uncertainty in the prediction of other data points can be obtained by a proper uncertainty propagation method. The Bayesian update can, then, be used for updating the distribution of the prediction distribution of these selected FVs.

PART III: REFERENCES

This part lists all the references appeared in this thesis.

References

[1] Amari, Shun-ichi, and Si Wu. "Improving support vector machine classifiers by modifying kernel functions." *Neural Networks* 12.6 (1999): 783-789.

[2] Antkowiak, Michal. "Artificial Neural Networks vs. Support Vector Machines for Skin Diseases Recognition." *Master's thesis, Department of Computing Science, Umea University, Sweden* (2006).

[3] Baena-Garc á, Manuel, et al. "Early drift detection method." (2006).

[4] Balabin, Roman M., Ravilya Z. Safieva, and Ekaterina I. Lomakina. "Gasoline classification using near infrared (NIR) spectroscopy data: comparison of multivariate techniques." *Analytica Chimica Acta* 671.1 (2010): 27-35.

[5] Baudat, Gaston, and Fatiha Anouar. "Feature vector selection and projection using kernels." *Neurocomputing* 55.1-2 (2003): 21-38.

[6] Baraldi, Piero, Roozbeh Razavi-Far, and Enrico Zio. "Classifier-ensemble incremental-learning procedure for nuclear transient identification at different operational conditions." *Reliability Engineering & System Safety* 96.4 (2011): 480-488.

[7] Bauer, Eric, and Ron Kohavi. "An empirical comparison of voting classification algorithms: Bagging, boosting, and variants." *Machine learning* 36.1-2 (1999): 105-139.

[8] Bennett, Kristin P., and Olvi L. Mangasarian. "Robust linear programming discrimination of two linearly inseparable sets." *Optimization methods and software* 1.1 (1992): 23-34.

[9] Burges, Christopher JC. "A tutorial on support vector machines for pattern recognition." *Data mining and knowledge discovery* 2.2 (1998): 121-167.

[10] Biagetti, Tatiana, and Enrico Sciubba. "Automatic diagnostics and prognostics of energy conversion processes via knowledge-based systems."*Energy* 29.12 (2004): 2553-2572.

[11] Bond, Leonard J., et al. "Prognostics and life beyond 60 years for nuclear power plants." *Prognostics and Health Management (PHM), 2011 IEEE Conference on*. IEEE, 2011.

[12] Boser, Bernhard E., Isabelle M. Guyon, and Vladimir N. Vapnik. "A training algorithm for optimal margin classifiers." *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, 1992.

[13] Brzezinski, Dariusz, and Jerzy Stefanowski. "Reacting to different types of concept drift: The accuracy updated ensemble algorithm." *Neural Networks and Learning Systems, IEEE Transactions on* 25.1 (2014): 81-94.

[14] Butler, Karen L. "An expert system based framework for an incipient failure detection and predictive maintenance system." *Intelligent Systems Applications to Power Systems, 1996. Proceedings, ISAP'96., International Conference on*. IEEE, 1996.

[15] Cauwenberghs, Gert, and Tomaso Poggio. "Incremental and decremental support vector machine learning." *Advances in neural information processing systems* (2001): 409-415.

[16] Csat ó, Lehel, and Manfred Opper. "Sparse on-line Gaussian processes." *Neural computation* 14.3 (2002): 641-668.

[17] Camargo, L., and Takashi Yoneyama. "Specification of training sets and the number of hidden neurons for multilayer perceptrons." *Neural computation*13.12 (2001): 2673-2680.

[18] Chapelle, Olivier. "Training a support vector machine in the primal." *Neural Computation* 19.5 (2007): 1155-1178.

[19] Cherkassky, Vladimir, and Yunqian Ma. "Practical selection of SVM parameters and noise estimation for SVM regression." *Neural networks* 17.1 (2004): 113-126.

[20] Chen, Shuyan, Wei Wang, and Henk Van Zuylen. "Construct support vector machine ensemble to detect traffic incident." *Expert systems with applications* 36.8 (2009): 10976-10986.

[21] Chen, Wun-Hwa, Sheng-Hsun Hsu, and Hwang-Pin Shen. "Application of SVM and ANN for intrusion detection." *Computers & Operations Research* 32.10 (2005): 2617-2634.

[22] Chen, Wun-Hua, Jen-Ying Shih, and Soushan Wu. "Comparison of support-vector machines and back propagation neural networks in forecasting the six major Asian stock markets." *International Journal of Electronic Finance* 1.1 (2006): 49-67.

[23] Chen, Sheng, Colin FN Cowan, and Peter M. Grant. "Orthogonal least squares learning algorithm for radial basis function networks." *Neural Networks, IEEE Transactions on* 2.2 (1991): 302-309.

[24] Cortes, Corinna, and Vladimir Vapnik. "Support-vector networks." *Machine learning* 20.3 (1995): 273-297.

[25] Crammer, Koby, et al. "Online passive-aggressive algorithms." *The Journal of Machine Learning*

Research 7 (2006): 551-585.
[26] Cristianini, Nello, and John Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.
[27] Dey, S. "Free overall in circular channels with flat base: a method of open channel flow measurement." *Flow Measurement and Instrumentation* 13.5 (2002): 209-221.
[28] Dietterich, Thomas G. "Ensemble methods in machine learning." *Multiple classifier systems*. Springer Berlin Heidelberg, 2000. 1-15.
[29] Dong, Jian-xiong, L. Devroye, and Ching Y. Suen. "Fast SVM training algorithm with decomposition on very large data sets." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 27.4 (2005): 603-618.
[30] Drucker, Harris, S. Wu, and Vladimir N. Vapnik. "Support vector machines for spam categorization." *Neural Networks, IEEE Transactions on* 10.5 (1999): 1048-1054.
[31] Engel, Stephen J., et al. "Prognostics, the real issues involved with predicting life remaining." *Aerospace Conference Proceedings, 2000 IEEE*. Vol. 6. IEEE, 2000.
[32] Erdem, Zeki, et al. "Ensemble of SVMs for incremental learning." *Multiple Classifier Systems*. Springer Berlin Heidelberg, 2005. 246-256.
[33] Elsayed, Elsayed A. *Reliability engineering*. Wiley Publishing, 2012.
[34] Fine, Shai, and Katya Scheinberg. "Efficient SVM training using low-rank kernel representations." *The Journal of Machine Learning Research* 2 (2002): 243-264.
[35] Gao, Junbin, Daming Shi, and Xiaomao Liu. "Significant vector learning to construct sparse kernel regression models." *Neural Networks* 20.7 (2007): 791-798.
[36] Gardner, M. W., and S. R. Dorling. "Artificial neural networks (the multilayer perceptron)--a review of applications in the atmospheric sciences." *Atmospheric environment* 32.14-15 (1998): 2627-2636.
[37] Gilardi, Nicolas, et al. "Environmental and pollution spatial data classification with support vector machines and geostatistics." *Greece, ACAI* 99 (1999): 43-51.
[38] Girosi, Federico. "An equivalence between sparse approximation and support vector machines." *Neural computation* 10.6 (1998): 1455-1480.
[39] Goh, Anthony TC. "Seismic liquefaction potential assessed by neural networks." *Journal of Geotechnical engineering* 120.9 (1994): 1467-1480.
[40] Gomes, João Bártolo, et al. "Mining recurring concepts in a dynamic feature space." (2013): 1-1.
[41] Gold, Carl, Alex Holub, and Peter Sollich. "Bayesian approach to feature selection and parameter tuning for support vector machine classifiers." *Neural Networks* 18.5 (2005): 693-701.
[42] Guo, Guodong, Stan Z. Li, and Kap Luk Chan. "Face recognition by support vector machines." *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*. IEEE, 2000.
[43] Hidalgo, Hugo, Sonia Sosa León, and Enrique Gómez-Trevino. "Application of the kernel method to the inverse geosounding problem." *Neural networks* 16.3 (2003): 349-353.
[44] Hida, T. (1960). Canonical representations of Gaussian processes and their applications. Memoirs of the College of Science, University of Kyoto. *Series A: Mathematics*, 33(1), 109-155.
[45] Hines, J. W., et al. "Empirical methods for process and equipment prognostics." *Reliability and Maintainability Symposium*. 2007.
[46] Hines, J. W., and D. R. Garvey. "Data Based Fault Detection, Diagnosis and Prognosis of Oil Drill Steering Systems." *Maintenance and Reliability Conference. Knoxville, Tennessee, USA*. 2007.
[47] Hanna, M. A., A. Y. Chikhani, and M. M. A. Salama. "Thermal analysis of power cables in multi-layered soil. I. Theoretical model." *Power Delivery, IEEE Transactions on* 8.3 (1993): 761-771.
[48] Huang, Guang-Bin, Paramasivan Saratchandran, and Narasimhan Sundararajan. "An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks." *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 34.6 (2004): 2284-2292.
[49] Hoerl, Arthur E., and Robert W. Kennard. "Ridge regression: Biased estimation for nonorthogonal problems." *Technometrics* 12.1 (1970): 55-67.
[50] Hofmann, D., et al. "Free volume distributions in ultrahigh and lower free volume polymers: comparison between molecular modeling and positron lifetime studies." *Macromolecules* 35.6 (2002): 2129-2140.
[51] Hofmann, Thomas, Bernhard Schölkopf, and Alexander J. Smola. "Kernel methods in machine learning." *The annals of statistics* (2008): 1171-1220.
[52] Hofmann, Thomas, Bernhard Schölkopf, and Alexander J. Smola. "Kernel methods in machine learning." *The annals of statistics* (2008): 1171-1220.
[53] Holmberg K., Komonen K., Oedewald P., Peltonen M., Reiman T., Rouhiainen V., Tervo J., and Heino

P.. "Safety and reliability technology review." *Res Rep BTUO43-031209. VTT Industrial Systems*, Espoo, (2004).

[54] Hong, Qiao, Zhang Bo, and Wang Min. "Online support vector machine based on convex hull vertices selection." *IEEE transactions on neural networks and learning systems* 24.4 (2013): 593-609.

[55] Huang, Xin, and Liangpei Zhang. "An SVM ensemble approach combining spectral, structural, and semantic features for the classification of high-resolution remotely sensed imagery." *Geoscience and Remote Sensing, IEEE Transactions on* 51.1 (2013): 257-272.

[56] Imrie, C. E., S. Durucan, and A. Korre. "River flow prediction using artificial neural networks: generalisation beyond the calibration range." *Journal of Hydrology* 233.1 (2000): 138-153.

[57] Inman, Joseph Robert. "Resistivity inversion with ridge regression." *Geophysics* 40.5 (1975): 798-817.

[58] ISO13381-1, Condition monitoring and diagnostics of machines – prognostics - Part1: General guidelines. International Standard, ISO, 2004.

[59] Javed, Kamran, Rafael Gouriveau, and Noureddine Zerhouni. "Novel failure prognostics approach with dynamic thresholds for machine degradation."*Industrial Electronics Society, IECON 2013-39th Annual Conference of the IEEE*. IEEE, 2013.

[60] T. Joachims. Making large-scale support vector machine learning practical. In B. Sch ¨ olkopf, C. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods*, chapter 12, pages 169 ‑ 184. MIT Press, 1999.

[61] Joachims, Thorsten. *Making large-scale SVM learning practical*. No. 1998, 28. Technische Universität Dortmund, Sonderforschungsbereich 475: Komplexitätsreduktion in multivariaten Datenstrukturen, 1998.

[62] Judson, Richard, et al. "A comparison of machine learning algorithms for chemical toxicity classification using a simulated multi-scale data model." *BMC bioinformatics* 9.1 (2008): 241.

[63] Kacprzynski, G. J., et al. "Predicting remaining life by fusing the physics of failure modeling with diagnostics." *JOm* 56.3 (2004): 29-35.

[64] Karasuyama, Masayuki, and Ichiro Takeuchi. "Multiple incremental decremental learning of support vector machines." *Advances in neural information processing systems*. 2009.

[65] Katto, Y. "A physical approach to critical heat flux of subcooled flow boiling in round tubes." *International journal of heat and mass transfer* 33.4 (1990): 611-620.

[66] Kadlec, Petr, and Bogdan Gabrys. "Local learning ‑ based adaptive soft sensor for catalyst activation prediction." *AIChE Journal* 57.5 (2011): 1288-1301.

[67] Kivinen, Jyrki, Alexander J. Smola, and Robert C. Williamson. "Online learning with kernels." *Signal Processing, IEEE Transactions on* 52.8 (2004): 2165-2176.

[68] Khan, Javed, et al. "Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks." *Nature medicine* 7.6 (2001): 673-679.

[69] Kim, Wan Joo, Soon Heung Chang, and Byung Ho Lee. "Application of neural networks to signal prediction in nuclear power plant." *Nuclear Science, IEEE Transactions on* 40.5 (1993): 1337-1341.

[70] Kim, Dong Su, et al. "Uncertainty analysis of data-based models for estimating collapse moments of wall-thinned pipe bends and elbows." *Nucl. Eng. Technol* 44.3 (2012): 323-330.

[71] Kothamasu, Ranganath, Samuel H. Huang, and William H. VerDuin. "System health monitoring and prognostics–a review of current paradigms and practices." *Handbook of Maintenance Management and Engineering*. Springer London, 2009. 337-362.

[72] Kolter, Jeremy Z., and Marcus A. Maloof. "Using additive expert ensembles to cope with concept drift." *Proceedings of the 22nd international conference on Machine learning*. ACM, 2005.

[73] Kon, Mark A., and Leszek Plaskota. "Information complexity of neural networks." *Neural Networks* 13.3 (2000): 365-375.

[74] Kothare, Mayuresh V., et al. "Level control in the steam generator of a nuclear power plant." *Control Systems Technology, IEEE Transactions on* 8.1 (2000): 55-69.

[75] Fine, Shai, and Katya Scheinberg. "Efficient SVM training using low-rank kernel representations." *The Journal of Machine Learning Research* 2 (2002): 243-264.

[76] Kotsiantis, Sotiris B., Ioannis D. Zaharakis, and Panayiotis E. Pintelas. "Machine learning: a review of classification and combining techniques." *Artificial Intelligence Review* 26.3 (2006): 159-190.

[77] Krishnapuram, Balaji, et al. "Sparse multinomial logistic regression: Fast algorithms and generalization bounds." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 27.6 (2005): 957-968.

[78] Lahssuny, Yunis, and Waldemar Jędral. "Universal correlations for predicting complete pump performance characteristics." *Journal of Power Technologies* 90 (2004).

[79] L ázaro-Gredilla, Miguel, Steven Van Vaerenbergh, and I. Santamaria. "A Bayesian approach to tracking with kernel recursive least-squares." *Machine Learning for Signal Processing (MLSP), 2011 IEEE*

International Workshop on. IEEE, 2011.
[80] LeCun, Yann, et al. "Comparison of learning algorithms for handwritten digit recognition." *International conference on artificial neural networks*. Vol. 60. 1995.
[81] Lee, Sukhan, and Rhee M. Kil. "A Gaussian potential function network with hierarchically self-organizing learning." *Neural Networks* 4.2 (1991): 207-224.
[82] Lembessis, E., et al. "CASSANDRA: an on-line expert system for fault prognosis." *Proc. the 5th CIM Europe Conference on Computer Integrated Manufacturing*. 1989.
[83] Leao, Bruno P., and Takashi Yoneyama. "On the use of the unscented transform for failure prognostics." *Aerospace Conference, 2011 IEEE*. IEEE, 2011.
[84] Li, Jixin. "An empirical comparison between SVMs and ANNs for speech recognition." *The First Instructional Conference on Machine Learning, iCML-2003 http://www. cs. rutgers. edu/~ mlittman/courses/ml03/iCML03/papers/li. pdf*. 2003.
[85] Li, Y., et al. "Adaptive prognostics for rolling element bearing condition." *Mechanical systems and signal processing* 13.1 (1999): 103-113.
[86] Lin, Shih-Wei, et al. "Particle swarm optimization for parameter determination and feature selection of support vector machines." *Expert Systems with Applications* 35.4 (2008): 1817-1824.
[87] Lin, Chih-Jen, and Ruby C. Weng. "Simple probabilistic predictions for support vector regression." *National Taiwan University, Taipei* (2004).
[88] Liu, Jie, et al. "Nuclear power plant components condition monitoring by probabilistic support vector machine." *Annals of Nuclear Energy* 56 (2013): 23-33.
[89] Luo, Jianhui, et al. "An interacting multiple model approach to model-based prognostics." *Systems, Man and Cybernetics, 2003. IEEE International Conference on*. Vol. 1. IEEE, 2003.
[90] Maier, Holger R., and Graeme C. Dandy. "The use of artificial neural networks for the prediction of water quality parameters." *Water resources research* 32.4 (1996): 1013-1022.
[91] Mangasarian, Y-J. Lee OL, and W. H. Wolberg. "Breast cancer survival and chemotherapy: a support vector machine analysis." *Discrete Mathematical Problems with Medical Applications: DIMACS Workshop Discrete Mathematical Problems with Medical Applications, December 8-10, 1999, DIMACS Center*. Vol. 55. American Mathematical Soc., 2000.
[92] Mikhail Kanevski, and Michel Maignan. *Analysis and modelling of spatial environmental data*. Vol. 6501. EPFL press, 2004.
[93] Minku, Leandro L., and Xin Yao. "DDD: A new ensemble approach for dealing with concept drift." *Knowledge and Data Engineering, IEEE Transactions on* 24.4 (2012): 619-633.
[94] Minku, Leandro L., Allan P. White, and Xin Yao. "The impact of diversity on online ensemble learning in the presence of concept drift." *Knowledge and Data Engineering, IEEE Transactions on* 22.5 (2010): 730-742.
[95] Moraes, Rodrigo, Joao Francisco Valiati, and Wilson P. GaviãO Neto. "Document-level sentiment classification: An empirical comparison between SVM and ANN." *Expert Systems with Applications* 40.2 (2013): 621-633.
[96] Coble, Jamie, Ramuhalli, Pradeep, Bond, Leonard, Hines, J. Wesley, and Upadhyaya, Belle. "A review of prognostics and health management applications in nuclear power plants." *International journal of prognostics and health management (submitted)*.
[97] Mobley, R. Keith. An introduction to predictive maintenance. Butterworth-Heinemann, 2002.
[98] Mukherjee, Sayan, Edgar Osuna, and Federico Girosi. "Nonlinear prediction of chaotic time series using support vector machines." *Neural Networks for Signal Processing [1997] VII. Proceedings of the 1997 IEEE Workshop*. IEEE, 1997.
[99] Muller, K., et al. "An introduction to kernel-based learning algorithms." *Neural Networks, IEEE Transactions on* 12.2 (2001): 181-201.
[100] Müller, K-R., et al. "Predicting time series with support vector machines." *Artificial Neural Networks—ICANN'97*. Springer Berlin Heidelberg, 1997. 999-1004.
[101] Muhlbaier, Michael D., and Robi Polikar. "An ensemble approach for incremental learning in nonstationary environments." *Multiple classifier systems*. Springer Berlin Heidelberg, 2007. 490-500.
[102] MacKay, David JC. "Gaussian processes-a replacement for supervised neural networks?" (1997).
[103] Neal, Radford M. *Bayesian learning for neural networks*. Diss. University of Toronto, 1995.
[104] Na, Man Gyun, Jin Weon Kim, and In Joon Hwang. "Collapse moment estimation by support vector machines for wall-thinned pipe bends and elbows." *Nuclear engineering and design* 237.5 (2007): 451-459.
[105] Na, Man Gyun, et al. "Prediction of major transient scenarios for severe accidents of nuclear power

plants." *Nuclear Science, IEEE Transactions on*51.2 (2004): 313-321.
[106] Neter, John, William Wasserman, and Michael H. Kutner. "Applied linear regression models." (1989).
[107] Ni, Jianjun, Chuanbiao Zhang, and Simon X. Yang. "An adaptive approach based on KPCA and SVM for real-time fault diagnosis of HVCBs." *Power Delivery, IEEE Transactions on* 26.3 (2011): 1960-1971.
[108] Nishida, Kyosuke, Koichiro Yamauchi, and Takashi Omori. "Ace: Adaptive classifiers-ensemble system for concept-drifting environments." *Multiple Classifier Systems*. Springer Berlin Heidelberg, 2005. 176-185.
[109] Nilsson, Nils. "Learning machines: Foundations of Trainable Pattern Classifying Systems." McGraw-Hill, 1965.
[110] Nguyen-Tuong, Duy, and Jan Peters. "Local gaussian process regression for real-time model-based robot control." *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, 2008.
[111] Oh, Sung-Kwun, Witold Pedrycz, and Byoung-Jun Park. "Polynomial neural networks architecture: analysis and design." *Computers & Electrical Engineering* 29.6 (2003): 703-725.
[112] Oppenheimer, Charles H., and Kenneth A. Loparo. "Physically based diagnosis and prognosis of cracked rotor shafts." *AeroSense 2002*. International Society for Optics and Photonics, 2002.
[113] Osuna, Edgar, Robert Freund, and Federico Girosi. "An improved training algorithm for support vector machines." *Neural Networks for Signal Processing [1997] VII. Proceedings of the 1997 IEEE Workshop*. IEEE, 1997.
[114] Palaniswami, M., et al. "Machine learning using support vector machines." *Proc. Int. Conf. Artificial Intelligence Science and Technology (AISAT)*. 2000.
[115] Page, E. S. "Continuous inspection schemes." *Biometrika* (1954): 100-115.
[116] Parikh, Devi, and Robi Polikar. "An ensemble-based incremental learning approach to data fusion." *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 37.2 (2007): 437-450.
[117] Peng, Ying, Ming Dong, and Ming Jian Zuo. "Current status of machine prognostics in condition-based maintenance: a review." *The International Journal of Advanced Manufacturing Technology* 50.1-4 (2010): 297-313.
[118] Pecht, Michael. "Prognostics and health monitoring of electronics."*Electronic Materials and Packaging, 2006. EMAP 2006. International Conference on*. IEEE, 2006.
[119] Platt, John. "Sequential minimal optimization: A fast algorithm for training support vector machines." (1998).
[120] Polat, Kemal, and Salih Güneş. "A new feature selection method on classification of medical datasets: Kernel F-score feature selection." *Expert Systems with Applications* 36.7 (2009): 10367-10373.
[121] Polikar, Robi. "Ensemble learning." *Ensemble Machine Learning*. Springer US, 2012. 1-34.
[122] Polikar, Robi, and Cesare Alippi. "Guest editorial learning in nonstationary and evolving environments." *IEEE transactions on neural networks and learning systems* 25.1 (2014): 9-11.
[123] Pozdnoukhov, A., and M. Kanevski. "Monitoring network optimisation using support vector machines." *Geostatistics for Environmental Applications*. Springer Berlin Heidelberg, 2005. 39-50.
[124] Rangwala, Huzefa, and George Karypis. "Profile-based direct kernels for remote homology detection and fold recognition." *Bioinformatics* 21.23 (2005): 4239-4247.
[125] Rasmussen, Carl Edward, and Zoubin, Ghahramani. "Infinite mixtures of Gaussian process experts." *Advances in neural information processing systems* 2 (2002): 881-888.
[126] Razavi-Far, Roozbeh, Piero Baraldi, and Enrico Zio. "Dynamic weighting ensembles for incremental learning and diagnosing new concept class faults in nuclear power systems." *Nuclear Science, IEEE Transactions on* 59.5 (2012): 2520-2530.
[127] Rasmussen, Carl Edward, and Zoubin Ghahramani. "Infinite mixtures of Gaussian process experts." *Advances in neural information processing systems* 2 (2002): 881-888.
[128] Rivas-Perea, Pablo, et al. "Support Vector Machines for Regression: A Succinct Review of Large-Scale and Linear Programming Formulations." *International Journal of Intelligence Science* 3 (2012): 5.
[129] Roh, Myung-Sub, Se-Woo Cheon, and Soon-Heung Chang. "Power prediction in nuclear power plants using a back-propagation learning neural network." *Nuclear technology* 94.2 (1991): 270-278.
[130] Roobaert, Danny, and Marc M. Van Hulle. "View-based 3d object recognition with support vector machines." *Neural Networks for Signal Processing IX, 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop.*. IEEE, 1999.
[131] Rosipal, Roman, and Leonard J. Trejo. "Kernel partial least squares regression in reproducing kernel hilbert space." *The Journal of Machine Learning Research* 2 (2002): 97-123.
[132] Suykens, Johan AK, and Joos Vandewalle. "Least squares support vector machine classifiers." *Neural*

processing letters 9.3 (1999): 293-300.
[133] Saunders, Craig, Alexander Gammerman, and Volodya Vovk. "Ridge regression learning algorithm in dual variables." *(ICML-1998) Proceedings of the 15th International Conference on Machine Learning*. Morgan Kaufmann, 1998.
[134] Schwabacher, Mark. "A survey of data-driven prognostics." *Proceedings of the AIAA Infotech@ Aerospace Conference*. 2005.
[135] Schölkopf, Bernhard, Ralf Herbrich, and Alex J. Smola. "A generalized representer theorem." *Computational learning theory*. Springer Berlin Heidelberg, 2001.
[136] Schölkopf, Bernhard, Alexander Smola, and Klaus-Robert Müller. "Nonlinear component analysis as a kernel eigenvalue problem." *Neural computation*10.5 (1998): 1299-1319.
[137] Scholkopf, Bernhard, et al. "Input space versus feature space in kernel-based methods." *Neural Networks, IEEE Transactions on* 10.5 (1999): 1000-1017.
[138] Schölkopf, Bernhard, et al. "Estimating the support of a high-dimensional distribution." *Neural computation* 13.7 (2001): 1443-1471.
[139] Schölkopf, Bernhard, Alexander Smola, and Klaus-Robert Müller. "Kernel principal component analysis." *Artificial Neural Networks—ICANN'97*. Springer Berlin Heidelberg, 1997. 583-588.
[140] Scholkopft, Bernhard, and Klaus-Robert Mullert. "Fisher discriminant analysis with kernels." *Neural networks for signal processing IX* (1999).
[141] Smola, Alex J., and Bernhard Schölkopf. "A tutorial on support vector regression." *Statistics and computing* 14.3 (2004): 199-222.
[142] Soares, Symone Gomes, and Rui Araújo. "An on-line weighted ensemble of regressor models to handle concept drifts." *Engineering Applications of Artificial Intelligence* 37 (2015): 392-406.
[143] Specht, Donald F. "Probabilistic neural networks." *Neural networks* 3.1 (1990): 109-118.
[144] Simonen, Fredric A., and Stephen R. Gosselin. "Life prediction and monitoring of nuclear power plant components for service-related degradation." *Journal of pressure vessel technology* 123.1 (2001): 58-64.
[145] Singh, Gajendra, et al. "A machine learning approach for detection of fraud based on svm." *International Journal of Scientific Engineering and Technology* 1.3: 194.
[146] Street, W. Nick, and YongSeog Kim. "A streaming ensemble algorithm (SEA) for large-scale classification." *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001.
[147] Support Vector Machines vs Artificial Neural Networks, http://www.svms.org/anns.html.
[148] Tao, Dacheng, et al. "Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 28.7 (2006): 1088-1099.
[149] Tikhonov, Andrej Nikolaevich, and Vasiliy Yakovlevich Arsenin. " Solutions of ill-posed problems." (1977).
[150] Tipping, Michael E. "Sparse Bayesian learning and the relevance vector machine." *The journal of machine learning research* 1 (2001): 211-244.
[151] Tran, Q-L., et al. "An empirical comparison of nine pattern classifiers." *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 35.5 (2005): 1079-1091.
[152] Tsang, Ivor W., James T. Kwok, and Pak-Ming Cheung. "Core vector machines: Fast SVM training on very large data sets." *Journal of Machine Learning Research*. 2005.
[153] Tsoukalas, Lefteri H., and Robert E. Uhrig. *Fuzzy and neural approaches in engineering*. John Wiley & Sons, Inc., 1996.
[154] Tu, Jack V. "Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes." *Journal of clinical epidemiology* 49.11 (1996): 1225-1231.
[155] Upadhyaya, Belle R., and Evren Eryurek. "Application of neural networks for sensor validation and plant monitoring." *Nuclear Technology* 97.2 (1992): 170-176.
[156] Van Gestel, Tony, et al. *Least squares support vector machines*. Vol. 4. Singapore: World Scientific, 2002.
[157] W.D. van Driel and X.J. Fan (eds.), Solid State Lighting Reliability: Components to Systems, Solid State Lighting Technology and Application Series 1, Springer Science+Business Media, LLC 2013.
[158] Vanajakshi, Lelitha, and Laurence R. Rilett. "A comparison of the performance of artificial neural networks and support vector machines for the prediction of traffic speed." *Intelligent Vehicles Symposium, 2004 IEEE*. IEEE, 2004.
[159] Valentini, Giorgio, and Thomas G. Dietterich. "Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods." *The Journal of Machine Learning Research* 5 (2004):

725-775.
[160] Valentini, Giorgio, and Thomas G. Dietterich. "Low bias bagged support vector machines." *ICML*. 2003.
[161] Vapnik, Vladimir Naumovich, and Vlamimir Vapnik. *Statistical learning theory*. Vol. 2. New York: Wiley, 1998.
[162] Vapnik, Vladimir. *The nature of statistical learning theory*. springer, 2000.
[163] Vichare, Nikhil, Brian Tuchband, and Michael Pecht. "Prognostics and Health Monitoring of Electronics." *Handbook of Performability Engineering*. Springer London, 2008. 1107-1122.
[164] Vachtsevanos, G. E. O. R. G. E., et al. "Intelligent fault diagnosis and prognosis for engineering systems, 2006." *Usa 454p Isbn* 13: 978-0.
[165] D. Wang et al., *Model-based Health Monitoring of Hybrid Systems*, Springer Science+Business Media New York 2013.
[166] Wang, Peng, and George Vachtsevanos. "Fault prognostics using dynamic wavelet neural networks." *AI EDAM* 15.04 (2001): 349-365.
[167] Wang, Di, et al. "An online core vector machine with adaptive MEB adjustment." *Pattern Recognition* 43.10 (2010): 3468-3482.
[168] Wang, Haixun, et al. "Mining concept-drifting data streams using ensemble classifiers." *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2003.
[169] Wang, Jack, Aaron Hertzmann, and David M. Blei. "Gaussian process dynamical models." *Advances in neural information processing systems*. 2005.
[170] Williams, Christopher KI. "Computing with infinite networks." *Advances in neural information processing systems* (1997): 295-301.
[171] Xu, Yong, et al. "A method for speeding up feature extraction based on KPCA." *Neurocomputing* 70.4 (2007): 1056-1061.
[172] Yager, Ronald R., and Lotfi Asker Zadeh, eds. *An introduction to fuzzy logic applications in intelligent systems*. Boston: Kluwer Academic, 1992.
[173] Yang, Jian, et al. "Two-dimensional PCA: a new approach to appearance-based face representation and recognition." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 26.1 (2004): 131-137.
[174] Yang, Heon Young, Man Gyun Na, and Jin Weon Kim. "Fuzzy support vector regression model for the calculation of the collapse moment for wall-thinned pipes." *Nuclear Engineering and Technology* 40.7 (2008): 607-614.
[175] Yang, Xinzhu, Bo Yuan, and Wenhuang Liu. "Dynamic Weighting ensembles for incremental learning." *Proc. of IEEE conference in pattern recognition*. 2009.
[176] Yegnanarayana, B. *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.
[177] Zliobaite, Indre, et al. "Active learning with drifting streaming data." *IEEE transactions on neural networks and learning systems* 25.1 (2014): 27-39.
[178] Zio, Enrico. "Prognostics and health management of industrial equipment." *Diagnostics and Prognostics of Engineering Systems: Methods and Techniques* (2012): 333-356.
[179] Zio, Enrico, and Francesco Di Maio. "A data-driven fuzzy approach for predicting the remaining useful life in dynamic failure scenarios of a nuclear system." *Reliability Engineering & System Safety* 95.1 (2010): 49-57.
[180] Zio, Enrico, Francesco Di Maio, and Marco Stasi. "A data-driven approach for predicting failure scenarios in nuclear systems." *Annals of Nuclear Energy* 37.4 (2010): 482-491.
[181] Zio, Enrico, Piero Baraldi, and Giulio Gola. "Feature-based classifier ensembles for diagnosing multiple faults in rotating machinery." *Applied Soft Computing* 8.4 (2008): 1365-1380.
[182] Zhang, Guoqiang, B. Eddy Patuwo, and Michael Y Hu. "Forecasting with artificial neural networks:: The state of the art." *International journal of forecasting* 14.1 (1998): 35-62.
[183] Zhao, W., Tao, T., & Zio, E. (2013a). System reliability prediction by support vector regression with analytic selection and genetic algorithm parameters selection. *Applied Soft Computing*, (submitted).
[184] Wei, Zhao, et al. "A dynamic particle filter-support vector regression method for reliability prediction." *Reliability Engineering & System Safety* 119 (2013): 109-116.
[185] Zhao, W., Tao, T., Zio, E. (2014). A Novel Hybrid Method of Parameters Tuning in Support Vector Regression for Reliability Prediction: Particle Swarm Optimization Combined with Analytical Selection (under submission).
[186] Zhu, Ji, and Trevor Hastie. "Kernel logistic regression and the import vector machine." *Journal of Computational and Graphical Statistics* 14.1 (2005).

[187] 舒梅, 张景玉, 廖隆源. "核电站主泵质量保证及核安全文化." *东方电机* 34.2 (2006): 62-67.

PART IV: PUBICATIONS

This part lists all the papers published or submitted to international journals.

PAPER I: JIE LIU, REDOUANE SERAOUI, VALERIA VITELLI & ENRICO ZIO "NUCLEAR POWER PLANT COMPONENTS CONDITION MONITORING BY PROBABILISTIC SUPPORT VECTOR MACHINE," *ANNALS OF NUCLEAR ENERGY*, VOL. 56, PP. 23-33, 2013.

NUCLEAR POWER PLANT COMPONENTS CONDITION MONITORING BY PROBABILISTIC SUPPORT VECTOR MACHINE

Jie Liu, Redouane Seraoui, Valeria Vitelli, Enrico Zio

ABSTRACT

In this paper, an approach for the prediction of the condition of Nuclear Power Plant (NPP) components is proposed, for the purposes of condition monitoring. It builds on a modified version of the Probabilistic Support Vector Regression (PSVR) method, which is based on the Bayesian probabilistic paradigm with a Gaussian prior. Specific techniques are introduced for the tuning of the PSVR hyerparameters, the model identification and the uncertainty analysis. A real case study is considered, regarding the prediction of a drifting process parameter of a NPP component.

Key words: Probabilistic support vector machine, Condition monitoring, Nuclear power plant, Point prediction

1. INTRODUCTION

Production systems are becoming increasingly complex and demand sophisticated methods to anticipate, diagnose and control abnormal events in a timely manner, as the consequences of unexpected faults can bring high economic losses for a company (Venkatasubramanian, 2005).

For an optimized operation, the conditions of NPP components and systems are usually monitored at regular intervals (Condition Monitoring), and a warning is triggered when the monitored signals exceed predefined thresholds (Fault Detection) (Zio et al., 2010).The plant operators must identify the plant state and the components out of control (Diagnostics), and predict the future development of the scenarios (Prognostics) to decide the actions to take to regain safe control of the plant (Zio, 2012). Then, while diagnostics aims at identifying the cause of the deviation from normal behavior and at determining the state of the parameters critical for the plant operation and safety, prognostics aims at the prediction of the Residual Useful Life (RUL) of the components (Zio, 2012).

In general, two strategies for condition monitoring, detection, diagnostics and prognostics are possible: either based on physical models, or based on data-driven approaches (Zio, 2012; Ma and Jiang, 2011). In the case of complex systems, physical models can be built only after simplification of the physical relations. Then, in most cases, they cannot timely provide the plant operators with a sufficiently precise diagnostics of the plant situation (Zio, 2012). On the contrary, data-driven approaches are attractive for NPPs, also considering that most components are monitored since the commissioning of plants, and, hence, a large amount of measured data is available to drive the tuning of the models (Ma and Jiang, 2011).

A substantial amount of research has concerned the development of data-driven approaches for condition monitoring, detection, diagnostics and prognostics. Artificial Neural Network (ANN), Support Vector Machine (SVM), Genetic Algorithm (GA) and Auto-Associative Kernel Regression (AAKR) are among some of the most studied and applied (Chevalier et al., 2009; Baraldi et al., 2010; Baradi et al., 2011; Santosh et al., 2009; Li et al., 2012; Yazikov et al., 2012; Rand et al., 2012a; Rand et al., 2012b; Muralidharan and Sugumaran, 2012; Ekici, 2012; Zio and Gola, 2006; Lu and Upadhyaya, 2005; Jeong et al., 2003; Zio et al., 2009). These approaches are already mature, especially for detection and diagnostics. On the contrary, the amount of research dealing with prognostics is limited, especially in the context of NPP components. Some recent references, referring to prognostics for engineering systems, are Li and Nilkitsaranont (2009), Niu and Yang (2010) and Wang et al. (2004). Support Vector

Regression (SVR) is used in Trontl et al. (2007) and Bae et al. (2008) to fulfill the point estimation with satisfactory results. In Elnokity et al. (2012), a hybrid modeling combined with the Industrial Source Complex (ISC) model and an Adaptive Neuro-Fuzzy Inference System (ANFIS) has been used to improve the modeling ability of predicting tracer concentrations. SVR method is used in Cai (2012) to predict the critical heat flux, while Fuzzy Neural Networks are used in Na et al. (2006) to estimate the collapse moment due to the wall-thinned defects of bends and elbows in piping systems. However, uncertainty quantification is not included in the previously described data-driven models. In Zio et al. (2010) and Zio and Di Miao (2010), a fuzzy similarity analysis is introduced to compare the evolving failure scenario with a library of reference patterns describing the multidimensional evolution of monitored process variables. The aim is to find a combination of the reference patterns, weighed by their similarity to the observed failure pattern, to determine the future evolution of the scenario and to derive the corresponding RUL. However, failure patterns in NPP components are rare and thus a "solid" library of references cannot be easily formed. SVR has also been used in Kim et al. (2012) to build Prediction Intervals (PIs) for the same problem. However, since the method has been trained on a relatively small amount of data, its generalization power is not assured.

It is well recognized that there exists no prognostic method that is ideal for every situation (Jardine et al., 2006; Y-C and Pepyne, 2001). A variety of methods have been developed for specific situations or specific classes of systems. In the present work, we propose a method for prediction with uncertainty quantification, in the context of NPP components condition monitoring and prognostics. We address the problem of predicting process variables under conditions of fault of a NPP component. A modified Probabilistic Support Vector Regression (PSVR) is developed and used to provide in output the PIs of a process variable. To the author's knowledge, this is the first time that such technique is applied in the specific application context of interest. A real case study is considered, related to the condition monitoring of a component of a NPP of Électricité De France (EDF). A main challenge arises from the need of building a model based on only one scenario, which is a realistic situation given the rarity of faults in NPP components.

The paper is structured as follows. Section 2 provides a description of the PSVR method for prognostics. Section 3 presents the characteristics of the data of the real case study, and the pre-treatment techniques used to remove the outliers, to reconstruct the missing data points, and to identify the most proper model. In Section 4, the results of the application of PSVR for prognostics are presented, and comparisons with the standard SVR method and other empirical

apporaches are also given in this Section. Some conclusions are drawn in Section 5.

2. PROBABILISTIC SUPPORT VECTOR REGRESSION (PSVR)

Standard Support Vector Machines (SVMs) (Cortes and Vapnik, 1995; Vapnik et al.; 1996; Boser et al., 1992; Drucker et al., 1997; Cristianini and Taylor, 2000) are learning machines implementing the Structural Risk Minimization (SRM) inductive principle to obtain good generalization performance on a limited number of learning patterns (Gao et al., 2002; Jardine et al., 2006; Poggio and Girosi, 1998; Girosi, 1998). However, the parameters need to be specifically tuned for the problem at hand and this may be difficult. Another problem related to SVMs is that the classification and regression results are provided as point estimates only, while it would be more informative to obtain a Prediction Interval (PI) with an associated probability that the true value lies in the interval. Also, the distribution of the predicted value is a constructive indicator for practical purposes.

To overcome these limitations, the Bayesian probabilistic paradigm has been considered in combination with SVM (Mackay, 1997; Neal, 1996; Williams, 1997). Recently, it has been shown that SVMs can be interpreted as a Maximum A Posteriori (MAP) solution to a Bayesian inference problem with Gaussian priors and an appropriate likelihood function. This probabilistic interpretation enables Bayesian methods to be employed to determine the regularization parameters in the SVM framework (Kim et al., 2012; Sollich, 1999). The method using MAP for SVM estimation is called Probabilistic Support Vector Regression (PSVR). Bayesian approaches for SVM can estimate the parameter and feature spaces simultaneously by maximizing the evidence function, and they allow obtaining an error bar for the prediction (Lin and Weng, 2004).

2.1 PSVR Using Ɛ-Insensitive Loss Function

Let us assume that the input data is a n-dimensional set of vectors $\boldsymbol{X} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \dots, \boldsymbol{x_n}\}$, independently drawn in $\boldsymbol{R}^p$, and that we also have an independent sample from the target value $\boldsymbol{Y} = \{y_1, y_2, \dots, y_n\}$, where $y_i \in \boldsymbol{R}, i = 1, 2, \dots, n$.

In regression methods, the final aim is to find an underlying function $a(\boldsymbol{x})$: $\boldsymbol{R}^p \to \boldsymbol{R}$ describing the relation between the input data and the target. We will now briefly state the PSVR approach to the estimation of $a(\boldsymbol{x})$; further mathematical details on the derivation of the method can be found in the Appendix, and in the references therein.

We make the following assumptions:

(5) Training data set $\boldsymbol{\Gamma} = \{\boldsymbol{X}, \boldsymbol{Y}\}$ follows an identical and independent distribution (i.i.d).

(6) The *a priori* probability distribution is $P[\boldsymbol{a}(\boldsymbol{X})] \propto \exp(-\frac{1}{2} \| \hat{P}a \|^2)$, where $\| \hat{P}a \|^2$ is a positive semi-definite operator and $\boldsymbol{a}(\boldsymbol{X}) = (a(\boldsymbol{x}_1), a(\boldsymbol{x}_2), \dots, a(\boldsymbol{x}_n))^T$.

(7) The $\mathcal{E}$ -insensitive loss function is chosen as the loss function.

(8) The covariance function is $K(\boldsymbol{x}, \boldsymbol{x}')$, and $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\frac{|x_i - x_j|^2}{2\gamma^2})$, where $\boldsymbol{x}_i$, $\boldsymbol{x}_j$ are the input data points in $\boldsymbol{X}$.

The a posteriori probability of $\boldsymbol{a}(\boldsymbol{X})$ can be written as

$$P[\boldsymbol{a}(\boldsymbol{X})|\boldsymbol{\Gamma}] = \frac{[G(C,\varepsilon)]^N}{\sqrt{det 2\pi K_{X,X}} P[\boldsymbol{\Gamma}]} \exp\{-C \sum_{\boldsymbol{x}_i \in \boldsymbol{X}} L_\varepsilon\big(y_i - a(\boldsymbol{x}_i)\big) - \frac{1}{2} \boldsymbol{a}(\boldsymbol{X})^T K_{\boldsymbol{X},\boldsymbol{X}}^{-1} \boldsymbol{a}(X)\}, \quad (1)$$

where $G(C, \varepsilon) = \frac{1}{2} \frac{C}{C\varepsilon + 1}$, and $K_{\boldsymbol{X},\boldsymbol{X}} = [K(\boldsymbol{x}_i, \boldsymbol{x}_j)]$ is the covariance matrix of the data points of $\boldsymbol{X}$.

We find the maximum of Equation (1) using the so-called MAP. This requires finding the minimum of the following function

$$R_{GSVM}(a) = C \sum_{\boldsymbol{x}_i \in \boldsymbol{X}} L_\varepsilon\big(y_i - a(\boldsymbol{x}_i)\big) + \frac{1}{2} \boldsymbol{a}(\boldsymbol{X})^T K_{\boldsymbol{X},\boldsymbol{X}}^{-1} \boldsymbol{a}(\boldsymbol{X}). \quad (2)$$

We can see that the risk of Gaussian SVMs is equivalent to the standard SVM. Following the discussion in Mackay (1997), Tikhonov and Arsenin (1997), Girosi (1998) and Burges (1998), we can write the solution of the minimization problem associated to Equation (2) in the following form

$$a^*(\boldsymbol{x}) = \sum_{\boldsymbol{x}_i \in \boldsymbol{X}} \beta_i K(\boldsymbol{x}_i, \boldsymbol{x}) \quad (3)$$

where $\beta_i = a_i - a_i^*$ is a combination of the Lagrange Multipliers associated to the optimization problem (Smola and Scholköpf, 2004). The a_i and a_i^* can be determined by a Quadratic Programming approach. According to Smola and Scholköpf (2004), $\forall\, i = 1, \dots, n$, a_i and a_i^* lie in the interval $[0, C]$, and β_i consequently lies in the interval $[-C, C]$, which is the domain of the optimization problem. See Na et al. (2006) for more details on the implementation.

2.2 Hyperparameters

According to the description of the PSVR method given in the previous Section, we shall now detail a strategy to determine the three hyperparameters C, ε, γ, before the optimization

algorithm is initialized.

Parameter C is the penalty factor. It controls the trade-off between complexity and the proportion of non-separable samples, and must be selected by the user (Vladimir et al., 1998). If it is too large, it will induce a high penalty for non-separable points, hence we may store too many support vectors and go towards over fitting. If it is too small, it may result in underfitting (Alpaydin, 2004). For what concerns the optimization process, C influences the computational burden of the regression: the bigger C is, the heavier the computational burden is.

Parameter ε controls the sparsity of the data. It has an effect on the smoothness of the SVM response and it affects the number of support vectors, so both the complexity and the generalization power of the network depend on its value (Horváth, 2001). By inspecting the ε-insensitive loss function (see the details in the Appendix), we see that data points inside a tube of radius ε surrounding the predicted values, are not considered in training the regression model. This is graphically exemplified in Figure 1.

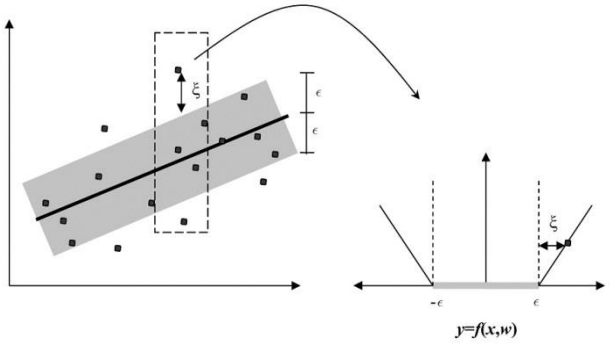

Fig.1 A picture of the ε-insensitive loss-function behavior.

Finally, parameter γ influences the width of the kernel, and hence the accuracy of the prediction and its variability.

There are already some methods in the literature to determine these hyperparameters, e.g. VC-theory in Vapnik (1995), Bayesian method in Mackay (1991), AIC in Akaike (1974), NIC in Murata et al. (1994) and Maximizing Evidence Function in Kim et al. (2012). In this paper, an interpolation method based on an innovative criterion is used to obtain the best values of these three parameters. The details are illustrated in Section 4, directly in relation to the case study.

2.3 Error Bar Estimation

In a Bayesian treatment of the prediction problem, error bars arise naturally from the predictive distribution. They are made up of two terms, one due to the *a posteriori* uncertainty (the uncertainty of $a(\boldsymbol{x})$), and the other due to the intrinsic noise in the data (Kim et al., 2012).

Suppose that $\boldsymbol{x}$ is a test input vector, and that the corresponding value of the target is the random variable y, obtained adding to $a(\boldsymbol{x})$ an unknown noise δ with zero mean; then

$$P[\boldsymbol{\Gamma}|\boldsymbol{a}(\boldsymbol{X})] \propto \exp(-C\sum_{i=1}^{n} l(\delta_i)). \quad (4)$$

We can also obtain the density of the noise δ

$$P[\delta] = \frac{C}{2(C\varepsilon+1)}\exp(-Cl_{\varepsilon}(\delta)), \quad (5)$$

and the noise variance

$$\sigma_{\delta}^2 = \frac{2}{C^2} + \frac{\varepsilon^2(C\varepsilon+3)}{3(C\varepsilon+1)}. \quad (6)$$

The conditional probability distribution of $a(\boldsymbol{x})$ given $\boldsymbol{\Gamma}$ can instead be written as

$$P[a(\boldsymbol{x})|\boldsymbol{\Gamma}] = \frac{1}{\sqrt{2\pi}\sigma_t}\exp\{-\frac{(a(x)-a^*(x))^2}{2\sigma_t^2}\}, \quad (7)$$

with

$$\sigma_t^2(\boldsymbol{x}) = K(\boldsymbol{x},\boldsymbol{x}) - K_{\boldsymbol{X}_M,\boldsymbol{x}}^T K_{\boldsymbol{X}_M,\boldsymbol{X}_M}^{-1} K_{\boldsymbol{X}_M,\boldsymbol{x}}. \quad (8)$$

Consequently, the error bar width of the prediction corresponding to the test input point $\boldsymbol{x}$ is

$$\sigma^2(\boldsymbol{x}) = \sigma_{\delta}^2 + \sigma_t^2(\boldsymbol{x}) = \frac{2}{C^2} + \frac{\varepsilon^2(C\varepsilon+3)}{3(C\varepsilon+1)} + K(\boldsymbol{x},\boldsymbol{x}) - K_{\boldsymbol{X}_M,\boldsymbol{x}}^T K_{\boldsymbol{X}_M,\boldsymbol{X}_M}^{-1} K_{\boldsymbol{X}_M,\boldsymbol{x}}. \quad (9)$$

The conditional probability distribution and the error bar are given in Equations (7) and (9). See Na et al. (2006) for more details on the calculations.

3. CASE STUDY DESCRIPTION

A set of data from the Reactor Coolant Pump (RCP) of one of EDF's NPPs is used to test the efficiency and the accuracy of the PSVR modeling approach developed in our work. In the following, we describe the data and illustrate the pre-processing steps.

3.1 Data Description

The dataset includes the measurements of the RCP of a NPP, with increasing leak flow in the first seal (a variable denoted with IntVar 9). The dataset contains the values of seventeen different variables recorded by seventeen different sensors along a period of 406 days. The variables whose measurements concern sensors inside the RCP are hereafter called internal variables; the others are called external variables. The description of all the internal and external variables and their physical meanings are given in Table 1.

There are nine internal variables and eight external variables. Each of the variables is observed hourly for a period of more than 13 months, about 9200 observation points. The evolution of four of the variables is shown in Figure 2: from left to right, and from top to bottom, ExtVar 2, ExtVar 6, ExtVar 7 and IntVar 9. At the 5700^{th} observation instance, we observe the fault, manifested by the variable IntVar 9 going out of control. We note that: all the variables are time-dependent, and there are seventeen variables in total, hence leading to a multivariate problem; each variable is measured hourly, giving 9205 measurements for each variable, and hence making computations challenging; all the variables show a nonlinear behavior, hence requiring a nonlinear model; the data need pre-processing, because there are many outliers and missing observations. Missing data are due to the absence of sensors recording during some time instances, while outliers correspond to bad (extremely high or low) sensor recordings. Concerning internal variables, the total number of missing data is 377 in IntVar 1, 415 in IntVar 2, 512 in IntVar 3, IntVar 4, IntVar 5 and IntVar 6, 493 in IntVar 7, 462 in IntVar 8 and 409 in IntVar 9. In the time series of external variables, there are 434 missing data in ExtVar 1, 409 in ExtVar 2, 422 in ExtVar 3, 428 in ExtVar 4, 372 in ExtVar 5, 453 in ExtVar 6, 512 in ExtVar 7, and 500 in ExtVar 8.

Tab.1 Physical meaning of each internal and external variable.

Internal variables		**External variables**	
Name	**Physical meaning**	**Name**	**Physical meaning**
IntVar 1	T cold leg loop 1 [WR]	ExtVar 1	T by-pass hot leg loop 3
IntVar 2	T water seal #1 051PO	ExtVar 2	T seal injection line
IntVar 3	T stator winding motor 051PO	ExtVar 3	P primary amount file B [GL]
IntVar 4	T motor lower bearing 051PO	ExtVar 4	Debit general file A
IntVar 5	T lower thrust bearing 051PO	ExtVar 5	Debit general file B
IntVar 6	T motor upper bearing 051PO	ExtVar 6	T aval exchanges file A
IntVar 7	T motor upper thrust bearing 051PO	ExtVar 7	T aval exchanges file B
IntVar 8	Flow seal injection supply RCP051PO	ExtVar 8	Debit refrigeration GMPP 051PO
IntVar 9	Seal leak flow #1 RCP051PO		

3.2 Data Pre-processing

Since the dataset we are going to analyze contains both missing data and outliers, we have to deal with both these issues. First of all, we will remove anomalous data, since their extreme values would affect the results of the analysis. Outliers can be easily detected by deciding some constraints, e.g. the limits $\bar{x} \pm 3 * \sigma_x$ where $\bar{x}$ is the mean of all the data points and σ_x is

their standard deviation. These limits are needed to detect the outliers, selected as those data points bigger than $\overline{x} + 3 * \sigma_x$ or smaller than$\overline{x} - 3 * \sigma_x$, and subsequently removed. Note that we used such constraints, rather than the usual ones based on the median and the InterQuartile Range (IQR), to be more conservative in the outlier selection, due to the dependence among data.

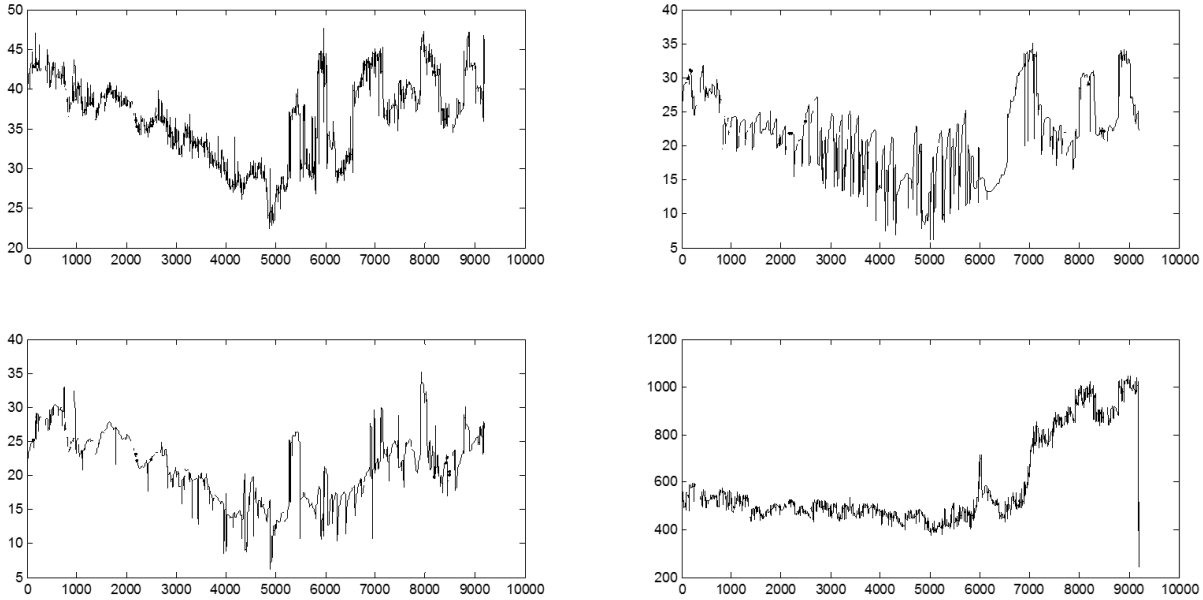

Fig.2 The evolution of four of the variables included in the dataset: from left to right, and from top to bottom, the measurements of ExtVar 2, ExtVar 6, ExtVar 7 and IntVar 9. On the axis, time measured in hours.

Secondly, we want to reconstruct missing data. Note that, after the outlier selection and elimination procedure, the number of missing data has increased. A possible way to deal with the reconstruction of missing data is the local polynomial regression fitting (Masry and Mielniczuk, 1999). This local least squares regression technique estimates effectively the values of the internal and external variables when there are missing data points. Moreover, it can also be used to perform the smoothing of the available observations, in order to reduce noise. We will thus use this technique both to reconstruct data where missing, and to obtain a smoother and less noisy time series in all remaining time instances.

Precisely, if we denote by t_0 a generic time instance, we execute the following steps to perform local polynomial regression:

(1) Find the k-nearest neighbors of t_0, which constitute a neighborhood $N(t_0)$: this means finding the k time instances in the time series which are closest to t_0. The number k is determined by setting it equal to a selected percentage (called *span*) of the

data; note that the *span* can be eventually different for each variable to allow flexibility. In the case of our application, three different values of the span have been selected, according to a trial-and-error procedure: 0.5% (high), 0.2% (medium) and 0.08% (low). For each of the variables, the most proper value of the *span* (high, medium or low) has been selected to be the most suited to the noise level of the variable.

(2) Calculate $D(t_0) = \max d(t, t_0)$ over $t \in N(t_0)$, where d is the Euclidean distance between the data at time t and t_0.

(3) For each point $t \in N(t_0)$, calculate its weight $W(t) = (1 - |\frac{t - t_0}{D(t_0)}|^3)^3$ with a tri-cube weight function.

(4) Calculate the weighted least square fit of t_0 on the neighborhood $N(t_0)$.

By repeating these steps for all time instances, all the variables are smoothed and reconstructed. Some examples of the so obtained time series are shown in Figure 3: they are the smoothed and reconstructed data corresponding to the variables in Figure 2. For the variables shown in Figure 3, the *span* parameter has been fixed to 0.5%.

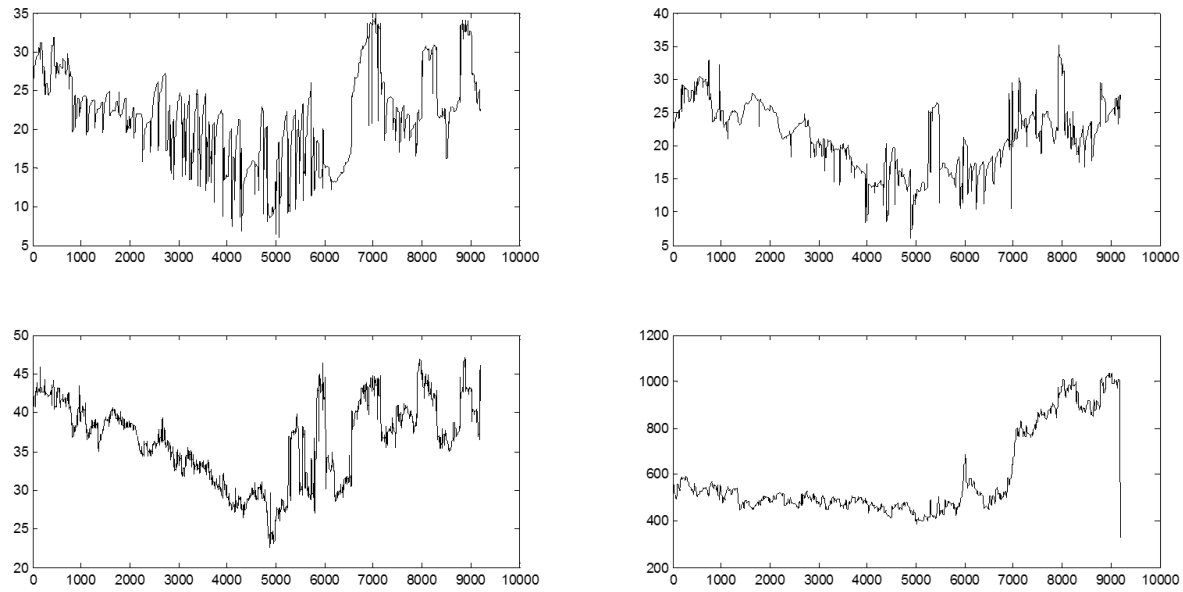

Fig.3 The smoothing and reconstruction of the evolution of the four variables whose raw observations are shown in Figure 2.

3.3 Model Identification

In order to select the most proper variables to be included as inputs in the PSVR model for improved prediction accuracy and reduction of the computational burden, a correlation analysis

is carried out between the target variable IntVar 9 and the other internal and external variables. The inputs are chosen to be the variables maximizing their correlations with the target IntVar 9. Correlations are measured by the classical Pearson correlation coefficient (Rodgers and Nicewander, 1988). Table 2 shows the results of the analysis.

Tab.2 Correlations of the target variable with other internal and external variables.

Correlations	**Internal variables**							
	IntVar 1	IntVar 2	IntVar 3	IntVar 4	IntVar 5	IntVar 6	IntVar 7	IntVar 8
IntVar 9	0.03128	0.48797	0.55268	0.50926	0.50701	0.58884	0.48164	0.19193
Correlations	**External variables**							
	ExtVar 1	ExtVar 2	ExtVar 3	ExtVar 4	ExtVar 5	ExtVar 6	ExtVar 7	ExtVar 8
IntVar 9	-0.44992	0.50569	0.12352	-0.24375	0.24569	0.43695	0.37322	-0.03361

Three external variables are the most related to the target: ExtVar 2, ExtVar 6 and ExtVar 7, corresponding to a correlation of 50.5%, 43.7% and 37.3%, respectively (see Table 2). Some of the internal variables have also a strong correlation with the target, with a correlation of more than 48%: IntVar 2, IntVar 3, IntVar 4, IntVar 5, IntVar 6 and IntVar 7. Hence, these six most related internal variables, and the three most related external variables, are included as inputs in the prediction model. IntVar 8 is also chosen as input, as suggested by expert judgment. The results are given in the next Section.

Historical values of the target can also be exploited as inputs to improve the accuracy of the prediction. In order to determine the most proper temporal horizon of the target for prediction purposes, i.e. the number of previous values to be used in the model, an autocorrelation analysis is carried out on the time series of the target values. The results of this analysis are reported in Figure 4, where the empirical partial autocorrelation function is plotted against the corresponding temporal lag (a multiple of one hour). It is evident that the correlations decrease with lag, and after a lag of three time steps (i.e. three hours) they are no longer significantly different from zero. Indeed, the dashed horizontal lines in the plot are the limits of the region of acceptance for a statistical test with null hypothesis being zero partial autocorrelation. Hence, only the first three historical values of the target, i.e. three hours before, are added as inputs to the three most correlated external variables.

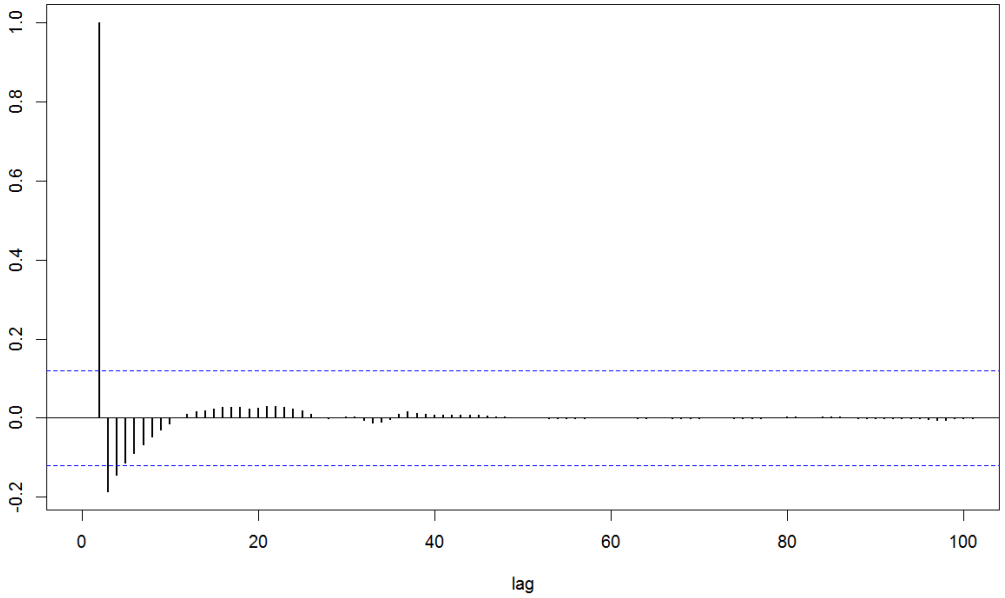

Fig.4 Empirical partial autocorrelation function of the time series of the target values (IntVar 9) with respect to time laps (multiples of hours)

4. CASE STUDY RESULTS

In this Section, we describe the results obtained using the PSVR method to give the prediction intervals for the target of interest in the context of condition monitoring of NPP components. The target is the variable IntVar 9, observed in its "out-of-control" regime after a fault occurred, and we focus on short-term (1-hour ahead) prediction. Assuming that we are at time t and we want to predict the target value at time $t+1$, we use as inputs the historical values of the target itself till three time steps before t, the values of the three most correlated external variables at time t and the values of the six most correlated internal variables at time t. We select a portion of the scenario under fault to apply PSVR for prediction: from the 5600th to the 8000th observed data. In this Section, 200 data points (5600th-5800th observations) are used for training and the rest for testing.

4.1 Tuning of the Hyperparameters

In order to achieve good prediction performance, we need to select the values of the hyperparameters C, ε and γ. The values of the hyperparameters influence the results of PSVR but a unifying method to determine their values has not yet been established. We propose a novel method which gives promising results. A comparison with two alternative methods is also conducted.

The method proposed in the present paper to determine the best values for the three hyperparameters is a simple but effective iterative search based on interpolation. Each

parameter is initially selected within a given interval. The best values are to be found by minimizing the following criterion

$$C_1 \sum_{i=1}^{n} \sigma_i + C_2 \sum_{i=1}^{n} |y_i^* - y_i| \quad (10)$$

where σ_i is the error bar width, $y_i^* = a^*(\boldsymbol{x}_i)$ the predicted value and y_i the target value of the i^{th} input data point. C_1 and C_2 are the two weights of the two parts of the objective function (Equation (10)), the error bar width and the bias of the prediction. If C_1 is smaller than C_2, it means that we pay more attention to the variance of the prediction (error bar width) than to the accuracy in the prediction (distance between target and predicted values), and vice versa for C_1 bigger than C_2. We fix $C \in [10, 10^5]$, $\gamma \in [10^{-7}, 10^3]$, $\varepsilon \in [10^{-3}, 10^{-1}]$, $C_1 = 4$ and $C_2 = 5$ by a trial-and-error process. For each parameter, a geometric sequence included in the corresponding interval is considered. In this applicative context, geometric sequences are better than arithmetic ones, since the parameter's influence on the objective function (Equation (10)) is highly non-linear. For C, ε and γ, geometric sequences of size 4, 10 and 4 are formed respectively. Note that for different training data sets, the best values of the parameters can change: hence, the tuning of the parameters in a feasible computational time is a relevant issue. In this case, the optimization of the objective function (Equation (10)) leads to the following choice for C, ε and γ: (6309.6, 0.0032, 7).

The results obtained via PSVR where the tuning of the hyperparameters is conducted according to the method proposed by the authors are compared with two alternative methods based on, respectively: the minimization of the objective function of the PSVR (Equation (2)) and the widely used minimization of the Mean Square Error (MSE) between the predicted value and the target of the training data set. The best combinations of C, ε and γ determined by these last two approaches are (398.1072, 0.3162, 2.5119) and (6309.6, 0.001, 3), respectively. A comparison of the results obtained with each of these strategies will be shown in the next Section.

4.2 PI for the Target and Conditional Predictive Distribution

The results of the application of PSVR are shown below. Figure 5 depicts the prediction of the target, with the corresponding Prediction Interval (PI) with a confidence level of 95%, obtained by tuning the hyperparmeters according to the novel strategy proposed by the authors. The solid line is the target, the dash-dot line is the point prediction, while the two dashed lines are the upper and lower bounds of the 95% PI computed according to the predictive distribution.

Hence, for each test point $\boldsymbol{x}$, the PI bounds are the values $L(\boldsymbol{x})$ and $U(\boldsymbol{x})$ corresponding to a 95% confidence that $y(\boldsymbol{x})$ lies in the interval $[L(\boldsymbol{x}), U(\boldsymbol{x})]$. In particular, the PI corresponding to the test point $\boldsymbol{x}$ is $[a^*(\boldsymbol{x}) - 2\sigma(\boldsymbol{x}), a^*(\boldsymbol{x}) + 2\sigma(\boldsymbol{x})]$, where $a^*(\boldsymbol{x})$ is the predicted value according to Equation (3) and $\sigma(\boldsymbol{x})$ is the variance associated to the prediction (error bar) and given by Equation (9). We remark that the predictive distribution in $\boldsymbol{x}$ is a Gaussian with mean $a^*(\boldsymbol{x})$ and variance $\sigma(\boldsymbol{x})$. Figure 6 shows the predictive distribution associated to the 7500th target data point according to Equation (7). The circle in Figure 6 is drawn in correspondence to the target value.

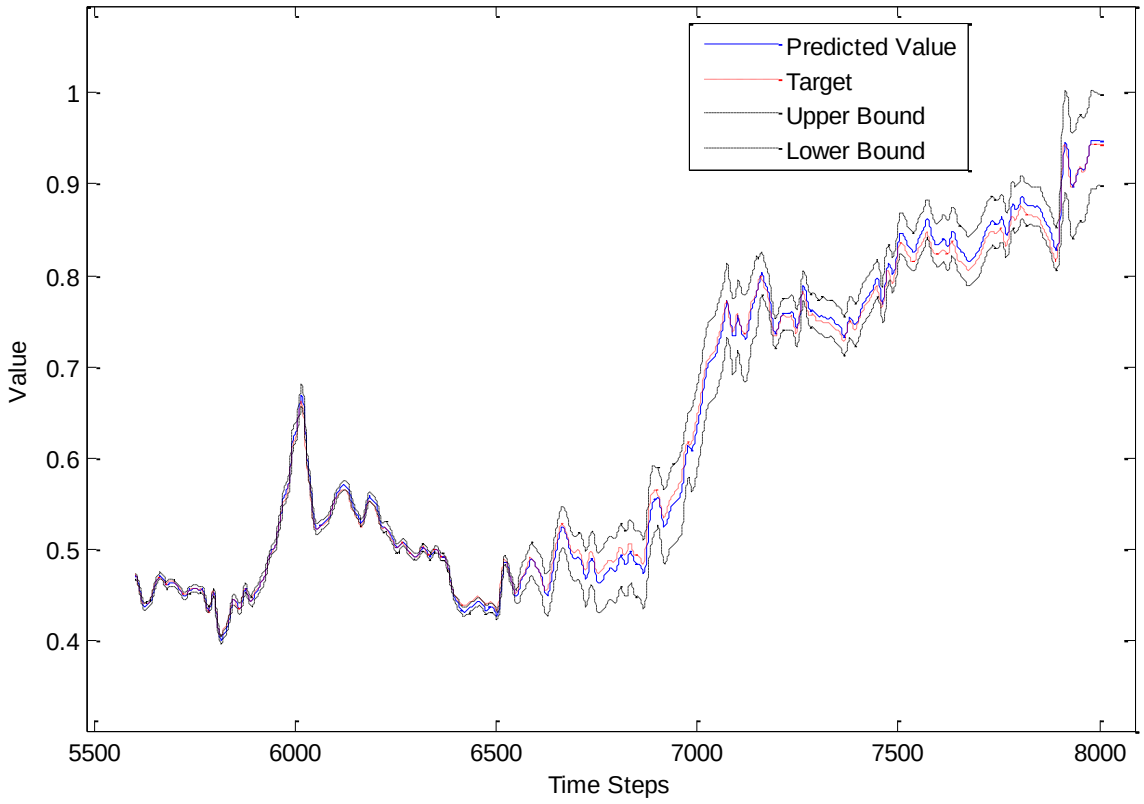

Fig.5 Point prediction and associated PIs for the target of interest (both the training and testing data points) using PSVR with hyperparameters tuning according to the proposed method.

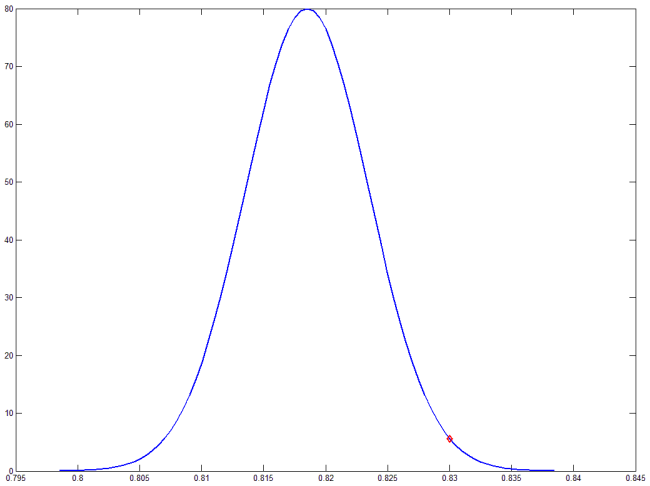

Fig.6 Predictive distribution associated to the 7500th target data point (circle), and obtained by using the PSVR with hyperparameters tuning according to the proposed method.

The prediction interval empirical coverage estimated on the whole testing set is 91.50%. The MSE is $5.4332*10^{-5}$. The relative error is smaller than 4%. If the model is trained using a bigger training data set, the relative error and absolute error will decrease.

The results of the application of PSVR (prediction of the target and 95% confidence PIs) with tuning of the hyperparameters according to the objective function (Equation (2)) and the MSE are shown in Figure 7 and Figure 8, respectively. Moreover a comparison of the three methods used for determining the values of the hyperparameters in terms of average width of PIs, mean relative error and mean absolute error is offered in Table 3. It is obvious that the proposed method is the best both in terms of prediction accuracy and precision. It is reasonable that the objective function of PSVR gives the worst results, because the objective function (Equation (2)) is used as a criterion to determine the weights of the support vectors in PSVR, and thus it is not expected to be suited also for determining the values of the hyperparameters. The results obtained via MSE are a little worse than the ones obtained by using the method proposed by the authors. This is mainly caused by the fact that MSE looks only to prediction accuracy and not at PIs width.

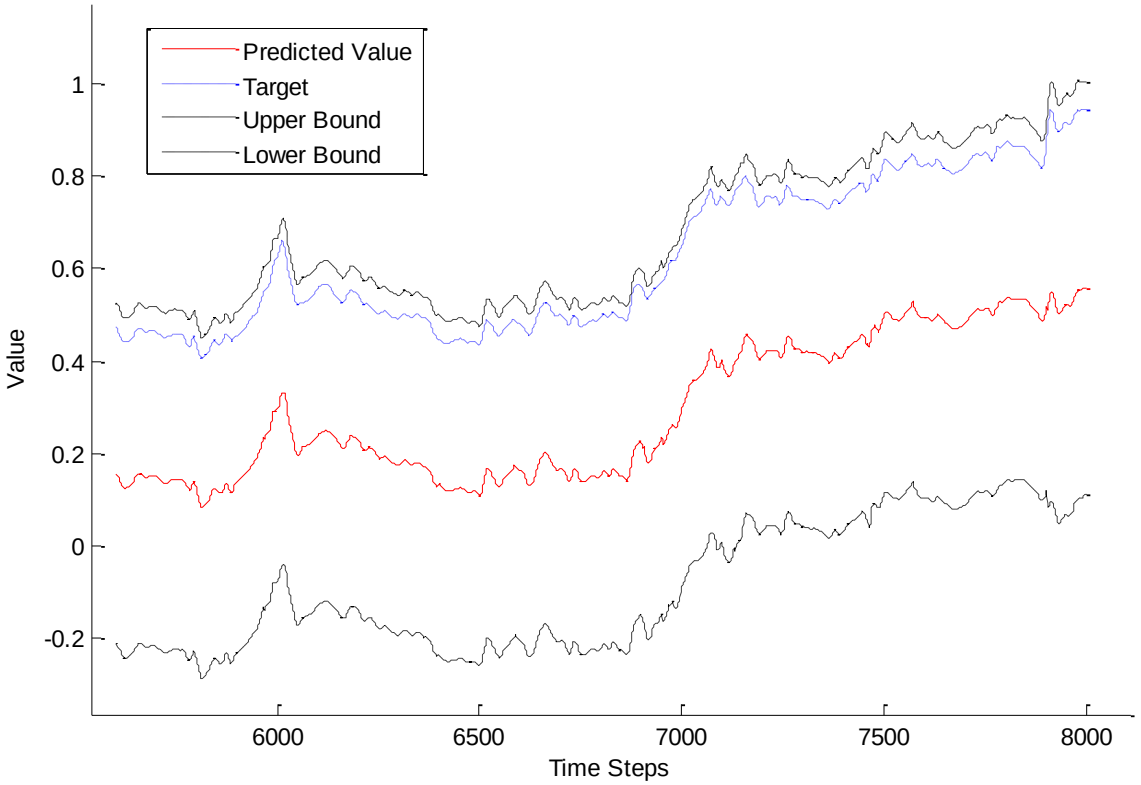

Fig.7 Point prediction and associated PIs for the target of interest (both the training and testing data points) using PSVR with hyperparameters tuning according to the objective function.

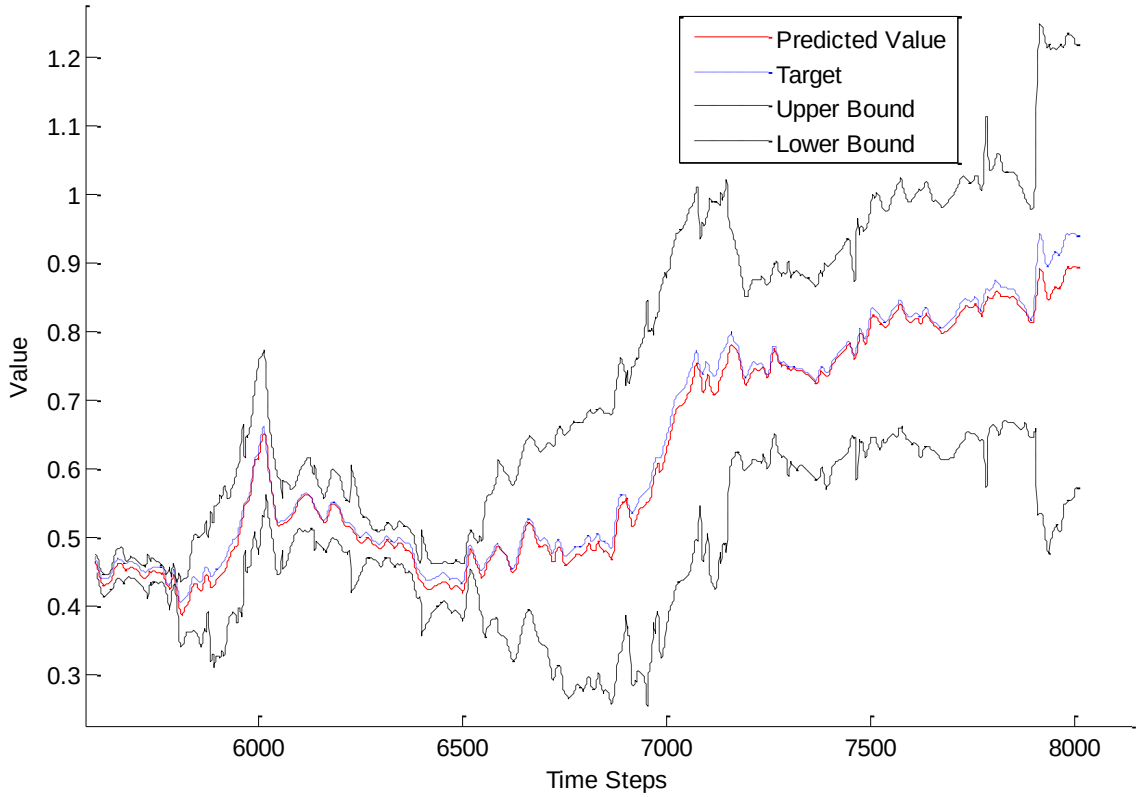

Fig.8 Point prediction and associated PIs for the target of interest (both the training and testing data points) using PSVR with hyperparameters tuning according to MSE.

Tab.3 Comparison of the results of different methods for determining the values of hyperparameters

Methods	Average Weights of PIs	Mean Relative Error	Mean Absolute Error
Proposed Method	0.0099	0.0093	0.0059
Objective Function of PSVR	0.1903	0.5613	0.3318
Mean Square Error	0.0671	0.0183	0.0114

4.3 Comparisons with other empirical approaches

In this Section, a comparison of the results obtained by PSVR and by other standard empirical approaches to short term prediction is illustrated. The empirical approaches we are going to consider are Auto-Associative Kernel Regression (AAKR) method, a well-established benchmark empirical approach to condition monitoring and prognostics, and Standard SVR, which corresponds to PSVR in a non-Bayesian framework.

AAKR is a well-known and established method suited both for reconstruction and for prediction purposes. It is an empirical modeling technique in which the prediction is found as a weighted sum of the previous values of the target variable. In order to determine the weights, AAKR makes use of historical observations of all signals to compute a global similarity measure, typically based on a Gaussian kernel, at each time. Further details on the method can be found in Baraldi *et al.* (2010). The main difference of this approach from both PSVR and SVR is the lack of a model for prediction: internal and external variables are used by AAKR just to compute the weights, but their patterns are not further exploited in the prediction process. On the contrary, both PSVR and SVR aim at finding the best non-linear empirical model relating the input variables (internal and external variables, and historical values of the target) to the future value of the target.

The comparison with AAKR has been carried out for three different training datasets, which correspond to the measurements intervals [6800^{th}, 6950^{th}], [6900^{th}, 7050^{th}] and [7000^{th}, 7150^{th}]. The bandwidth of the Gaussian kernel used in AAKR has been tuned for each dataset by a trial-and-error process, and the resulting best values are 2, 2 and 1, respectively. We show in Figure 9 the results obtained by AAKR on the second training dataset, [6900^{th}, 7050^{th}], where we trained AAKR on the same signals (internal and external variables) used as inputs in both PSVR

and SVR models: the solid line in the Figure is the target and the dashed line is the prediction given by AAKR. Note that the data normalization strategy used in the AAKR procedure is different from the one used in PSVR and SVR: in the former case, data have been normalized to have zero mean and standard deviation equal to 1, while in the latter case they have been forced to lie in the interval [0,1].

It is evident from Figure 9 that AAKR method does not give satisfactory results. Actually these poor results should be expected, since AAKR is in general proficiently used for condition monitoring and fault detection, but it is not a proper method for prognostics: it is capable of effectively reconstructing the operational behavior of a signal, but since it computes only a weighted average of the signals in the training set, its generalization power is low in the case of out-of-control signals.

For what concerns the comparison of PSVR with SVR, the SVM-Toolbox of Matlab is used. The comparison is carried out for the same three training datasets considered for the comparison with AAKR. Using the same values for the hyperparameters selected for PSVR, the result of SVR on the training dataset [6900th, 7050th] is shown in Figure 10, where the solid line is the target and the dashed line is the predicted value.

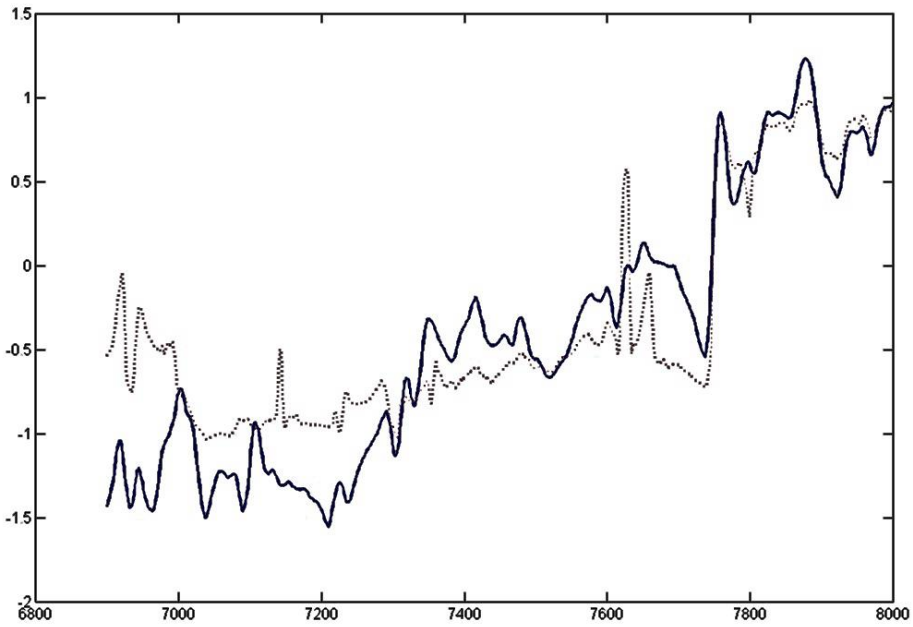

Fig.9 Point prediction for the target of interest (for both the training and testing data points) using AAKR (the bandwidth of the Gaussian kernel is set equal to 2).

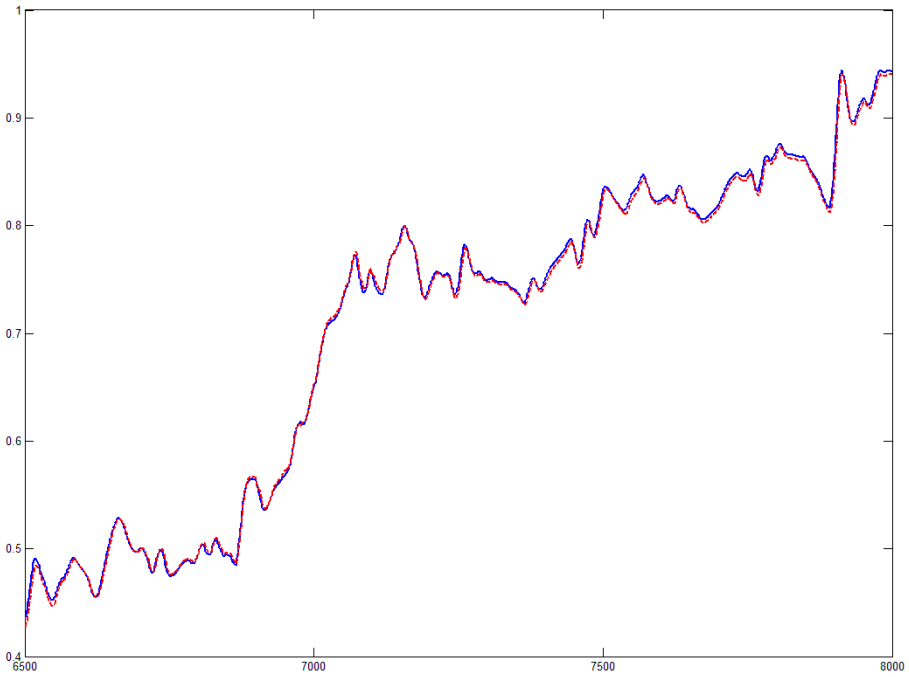

Fig.10 Point prediction for the target of interest (for both the training and testing data points) using standard SVR (with Matlab Toolbox).

Differently from the case of AAKR, the prediction obtained with SVR is quite accurate. Hence, to obtain a more precise comparison, in Table 4 the values of the Mean Square Errors obtained for the three training data sets with SVR and PSVR are reported. From inspection of the Table, it can be noticed that PSVR and standard SVR give comparable results, since the result of PSVR is slightly better for the first and third training data sets, but it is slightly worst for the second one. This is probably due to the empirical nature of the nonlinear regression methods we are using. On the other hand, standard SVR can only give a point estimate, while PSVR can also provide the uncertainty quantification, e.g. the PIs for the target and the predictive distribution.

Tab.4 Comparison of the results of PSVR and standard SVR

	SVR	PSVR
[6800th, 6950th]	$3.3353*10^{-4}$	$2.2267*10^{-5}$
[6900th, 7050th]	$1.4105*10^{-5}$	$5.3534*10^{-5}$
[7000th, 7150th]	$1.9787*10^{-4}$	$2.8867*10^{-5}$

5. CONCLUSION

In this paper, an approach is proposed for prediction of parameters of NPP components under fault conditions. It includes pre-processing for data reconstruction and model selection, and

PSVR for estimation of the prediction interval and conditional predictive distribution of the target of interest. The results of the application to a real case study of leak flow in the first seal of a RCP are satisfactory. The coverage of the prediction interval is 91.50% with a confidence level of 95%. The conditional predictive distribution provides the probability distribution of the values of the target. These two indicators, the PI and the predictive distribution, are very informative for the NPP operators in case of accident.

The future work will focus on the development of a method to extend condition monitoring to prognostics, by computing the NPP components RUL on the basis of the prediction of its evolving parameters. This entails propagating the uncertainties in the prediction, due to both the observed data and the model itself.

REFERENCES

H. Akaike, 1974. "A new look at the statistical model identification". IEEE Trans. Auto. Control. 19(6), PP.716–723.

E. Alpaydin, 2004. "Introduction to Machine Learning". The MIT Press. Cambridge, Massashusetss, London, England.

I.H. Bae, M.G. Na, Y.J. Lee, G.C. Park, 2008. "Calculation of the power peaking factor in a nuclear reactor using support vector regression models". Ann. Nucl. Energy. 35, PP.2200-2205.

P. Baraldi, R. Canesi, E. Zio, R. Seraoui and R. Chevalier, 2010. "Signal grouping for condition monitoring of nuclear power plant components". NPIC&HMIT 2010, Las Vegas, Nevada, USA, November.

P. Baraldi, R. Canesi, E. Zio, R. Seraoui and R. Chevalier, 2011."Genetic algorithm-based wrapper approach for grouping condition monitoring signals of nuclear power plant components". Integr. Comput. Eng. 18(3), PP. 221-234.

B.E. Boser, I.M. Guyon, V.N. Vapnik, 1992. "A training algorithm for optimal margin classifiers". 5th Annual ACM Workshop on COLT, Pittsburgh, PA, ACM Press, pp. 144-152.

C.J.C. Burges, 1998. "A Tutorial on Support Vector Machines for Pattern Recognition". Data Min. & Knowl. Discov. 2(2), PP.121-167.

J.J. Cai, 2012. "Applying support vector machine to predict the critical heat flux in concentric-tube open thermosiphon". Ann. Nucl. Energy. 43, PP.114-122.

R. Chevalier, D. Provost and R. Seraoui, 2009. "Assessment of statistical and classification models for monitoring EDF's assets". Sixth American Nuclear Society International Topical Meeting on Nuclear Plant Instrumentation, Control, and Human-Machine Interface Technologies, Knoxville, Tennessee, USA.

C. Cortes, V.N. Vapnik, 1995. "Support-Vector Networks". Machine Learning, vol. 20.

N. Cristianini and G. S. Taylor, 2000. "An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods", Cambridge University Press.

H. Drucker, C. J. C. B urges, L. Kaufman, A.J. Smola, V.N. Vapnik, 1997. "Support vector regression machines". Advances in Neural Information Processing Systems (NIPS 1996; MIT Press), 9, PP. 155-161.

S. Ekici, 2012. "Support Vector Machines for classification and locating faults on transmission lines". Appl. Soft Comp. 12, PP.1650-1658.

O. Elnokity, I.I. Mahmoud, M.K. Refai, H.M. Farahat, 2012. "ANN based Sensor Faults Detection, Isolation, and Reading Estimates –SFDIRE: Applied in a nuclear process". Ann. Nucl. Energy. 49, PP.131-142.

J.B. Gao, S.R. Gunn and C.J. HarrisetM. Brown, 2002. "A Probabilistic Framework for SVM Regression and Error Bar Estimation". Mach. Learn. 46(1-3), PP.71-89.

F. Girosi, 1998. "An equivalence between sparse approximation and support vector machines". Neural Comp. 10(6), PP.1455–1480.

G. Horváth. "Neural Networks in System Identification". In: V. Piuri (Ed.) Neural Networks in Measurement Systems NATO ASI NIMIA, Crema, Italy 2001. Oct. IOS Press, in print.

A.K.S. Jardine, D. Lin and D. Banjevic, 2006. "A review on machinery diagnostics and prognostics

implementing condition-based maintenance". Int. J. Adv. Manuf. 28(9-10), PP.1012-1024.
H. Jeong, W. Hwang, E. Kim and M. Han, 2003. "Hybrid modeling approach to improve the forecasting capability for the gaseous radionuclide in a nuclear site". Ann. Nucl. Energy. 30, PP.1365-138.
D.S. Kim, J.H. Kim, M. Gyunna and J.W. Kim, 2012. "Uncertainty analysis of data-based models for estimating collapse moments of wall-thinned pipe bends and elbows". Nucl. Eng. & Tech. 44(3), PP. 323-330.
F. Li, U.B.R. Perillo and S.R.P, 2012. "Fault Diagnosis of Helical Coil Steam Generator Systems of an Integral Pressurized Water Reactor Using Optimal Sensor Selection". IEEE Trans. Nucl. Sci. 59(2), PP. 403-410.
Y. G. Li and P. Nilkitsaranont, 2009. "Gas turbine performance prognostic for condition-based maintenance", Applied Energy, 86(10), PP. 2152–2161.
C.J. Lin and R.C. Weng, 2004. "Simple probabilistic predictions for support vector regression". National Taiwan University, Taipei.
B. Lu, B.R. Upadhyaya, 2005. "Monitoring and fault diagnosis of the steam generator system of a nuclear power plant using data-driven modeling and residual space analysis". Ann. Nucl. Energy. 32, PP.897-912.
J.P. Ma and J. Jiang, 2011. "Applications of fault detection and diagnosis methods in nuclear power plants: A review". Prog. Nucl. Energy, 53, PP.255-266.
D.J. MacKay, 1991. "Bayesian Modelling and Neural Networks". PhD thesis, California Institute of Technology, Pasadena, CA.
D.J. MacKay, 1997. "Gaussian processes, a replacement for neural networks". NIPS tutorial 1997, Cambridge University.
E. Masry and J. Mielniczuk, 1999. "Local linear regression estimation for time series with long-range dependence". Stoch. Process. & Appl. 82 (2), PP.173–193.
V. Muralidharan and V. Sugumaran, 2012. "A comparative study of Naïve Bayes classifier and Bayes net classifier for fault diagnosis of monoblock centrifugal pump using wavelet analysis". Appl. Soft Comp. 12, PP.2023-2029.
N. Murata, S. Yoshizawa and S. Amari, 1994. "Network information criterion–determining the number of hidden units for artificial neural network models". IEEE Trans. Netw. 5, PP.865–872.
M. G. Na, J. W. Kim, and D. N. Moreton, 2006. "Estimation of Collapse Moment for the Wall-Thinned Pipe Bends Using Fuzzy Model Identification". Nucl. Eng. & Des. 236, PP. 1335–1343.
R. Neal, 1996. "Bayesian Learning for Neural Networks". Lecture Notes in Statistics. Springer, New York.
G. Niu and B. S. Yang, 2010. "Intelligent condition monitoring and prognostics system based on data-fusion strategy", Expert Systems with Applications, 37(12), PP. 8831–8840.
T. Poggio and F. Girosi, 1998. "A sparse representation for function approximation". Neural Comp. 10, PP.1445–1454.
C.P. du Rand and G. van Schoor, 2012a. "Fault diagnosis of generation IV nuclear HTGR components – Part I: The error enthalpy–entropy graph approach". Ann. Nucl. Energy. 40, PP.14-24.
C.P. du Rand and G. van Schoor, 2012b. "Fault diagnosis of generation IV nuclear HTGR components – Part II: The error enthalpy–entropy graph approach". Ann. Nucl. Energy. 41, PP.79-86.
J.L. Rodgers and W.A. Nicewander, 1988. "Thirteen ways to look at the correlation coefficient". Am. Stat. 42 (1), PP. 59-66.
A.J. Smola and B. Schölkopf, 2004. "A tutorial on support vector regression". Stat. & Comp. 14(3), PP.199-222.
P. Sollich, 1999. "Probabilistic interpretations and Bayesian methods for support vector machines". Technical report, King's College London, London, UK.
A.N. Tikhonov and V.Y. Arsenin, 1977. "Solution of Ill-posed Problems". W.H. Winston, Washington, D.C..
K. Trontl, T. Smuc, D. Pevec, 2007. "Support vector regression model for the estimation of c-ray buildup factors for multi-layer shields". Ann. Nucl. Energy. 34, PP.939-952 (2007).
T. V. Santosh, A.Srivastava, V.V.S.SanyasiRao, A.K.Ghosh and H.S.Kushwaha, 2009. "Diagnostic system for identification of accident scenarios in nuclear power plants using artificial neural networks". Reliab. Eng. Syst. Saf. 94, PP. 759-762.
V.N. Vapnik, 1995. "The Nature of Statistical Learning Theory". Springer. New York.
V. N. Vapnik, S. E. Golowich and A. Smola, 1996. "Support vector method for function approximation, regression estimation, and signal processing", Proceedings of the 10th Neural Information Processing Systems (NIPS) Conference, Denver, Colorado.
V. Venkatasubramanian, 2005. "Prognostic and Diagnostic Monitoring of Complex Systems for Product Lifecycle Management: Challenges and opportunities". Comput. Chem. Eng. 29(6), PP. 1253-1263.

W. Q. Wang, M. F. Golnaraghi and F. Ismail, 2004. "Prognosis of machine health condition using neuro-fuzzy systems", Mechanical Systems and Signal Processing, 18(4), PP. 813–831.
C. Vladimir and F. Mulier, 1998. "Learning from Data: Concepts, Theory, and Methods". John Wiley camp; Sons, Inc. New York, N.Y., USA.
C.K. Williams, 1997. "Computing with infinite networks". In M.C. Mozer, M.I. Jordan, and T. Petsche, editors, Info. Process. Sys. volume 9, PP.295–301. MIT Press.
M. Yazikov., G. Gola , O. Berg, J. Porsmyr, H. Valseth, D. Roverso and M. Hoffmann, 2012. "On-Line Fault Recognition System for the Analogic Channels of VVER 1000/400 Nuclear Reactors". IEEE Trans. Nucl. Sci. 59(2), PP. 411-418.
H. Y-C and DL Pepyne, 2001. "Simple Explanation of the No Free Lunch Theorem of Optimization". In Proceedings of the 40th IEEE Conference on Decision and Control, 5 PP. 4409-4414. December 4-7, Orlando, Florida. IEEE, Piscataway, New Jersey.
E. Zio, 2012. "Diagnostics and Prognostics of Engineering Systems: Methods and Techniques", Chapter 17. Engineering Science Reference. USA.
E. Zio, F. Di Maio and M. Stasi, 2010. "A data-driven approach for predicting failure scenarios in nuclear systems", Ann. Nucl. Energy. 37(4), PP. 482-491.
E. Zio and F. Di Maio, 2010. "A data-driven fuzzy approach for predicting the remaining useful life in dynamic failure scenarios of a nuclear system". Reliab. Eng. Syst. Saf. 95, PP. 45-57.
E. Zio, G. Gola, 2006. "Neuro-fuzzy pattern classification for fault diagnosis in nuclear components". Ann. Nucl. Energy. 33, PP.415-426.
E. Zio, P. Baraldi, I. C. Popescu, 2009. "A fuzzy decision tree method for fault classification in the steam generator of a pressurized water reactor". Ann. Nucl. Energy. 36, PP.1159-1169.

APPENDIX A

Let us assume that the input data is a n-dimensional set of vectors $\boldsymbol{X} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \dots, \boldsymbol{x_n}\}$, independently drawn in $\boldsymbol{R}^p$, and that we also have an independent sample from the target value $\boldsymbol{Y} = \{y_1, y_2, \dots, y_n\}$, where $y_i \in \boldsymbol{R}, i = 1, 2, \dots, n$.

In regression methods, the final aim is to find an underlying function describing the relation between the input data and the target. Here, this function will be indicated as an element of the generic space

$$F = \{a(\boldsymbol{x}): \boldsymbol{R}^p \to \boldsymbol{R}\}. \quad (1)$$

Moreover, we assume that the training set $\boldsymbol{\Gamma} = \{\boldsymbol{X}, \boldsymbol{Y}\}$ has been drawn from the probability distribution $P(\boldsymbol{x}, \boldsymbol{y}): \boldsymbol{R}^{p+1} \to \boldsymbol{R}$, which is not known. The Maximum A Posteriori (MAP) method consists in finding the $a(\boldsymbol{x})$ which minimizes the risk

$$R_{ERM}(\boldsymbol{x}) = \int l(a(\boldsymbol{x}) - y) dP(\boldsymbol{x}, y), \quad (2)$$

where $l(\boldsymbol{x}, y)$ is the loss function. There are many possible choices for the loss functions, e.g. square loss function, 1-norm loss function, Huber's loss function, etc. In this paper, we adopt one of the most common choices, the ε-insensitive loss function

$$l(x) = \begin{cases} 0 & if\ |\boldsymbol{x}| < \varepsilon \\ |x| - \varepsilon & if |x| \geq \varepsilon \end{cases}. \quad (3)$$

The ε-insensitive loss function has a good sparseness property, because all the data points whose margin between the predicted and target values is smaller than ε, are not used in the estimation process.

In SVM, the Empirical Risk Minimization (ERM) is used to solve the optimization problem in Equation (2), where $dP(\boldsymbol{x}, y)$ is replaced by $\frac{1}{n}$, recalling that n is the number of input data points. This means assuming that all the data points follow an identical and independent distribution (i.i.d), and using their empirical sample distribution. However, according to Tikhonov and Arsenin (1977), this is an ill-posed approach, whose generalization property is not good.

The Structural Risk Minimization (SRM) is formulated to solve the problem. A positive semi-definite operator $\| \hat{P}a \|^2$ is added to the ERM, and the so obtained new risk functional, called SRM, is given by

$$R_{SRM}(\boldsymbol{x}) = C \sum_{\boldsymbol{x}_i} l(a(\boldsymbol{x}_i) - y_i) + \frac{1}{2} \| \hat{P}a \|^2. \quad (4)$$

The operator $\hat{\mathrm{P}}$ maps the space F into a dot-product space, and the kernel kernel $K = (\hat{P}^T \hat{P})^{-1}$ is derived after the Gaussian Process (GP) is introduced as a prior into the regression problem.

Indicating with $\boldsymbol{a}(\boldsymbol{X}) = (a(\boldsymbol{x}_1), a(\boldsymbol{x}_2), \ldots, a(\boldsymbol{x}_n))^T$ the vector of function values, $P[\boldsymbol{a}(\boldsymbol{X})|\boldsymbol{\Gamma}]$ is the conditional probability of $\boldsymbol{a}(\boldsymbol{x})$ given the training set $\boldsymbol{\Gamma}$. $P[\boldsymbol{\Gamma}|\boldsymbol{a}(\boldsymbol{X})]$ is the likelihood of $\boldsymbol{X}$ having been originated by the corresponding target $\boldsymbol{Y}$ given the underlying function $a(\boldsymbol{x})$. $P[\boldsymbol{a}(\boldsymbol{X})]$ is the a priori probability of the underlying function $a(\boldsymbol{x})$. $P[\boldsymbol{\Gamma}]$ is the evidence.

Applying the Bayesian Rule, we can derive the relation

$$P[\boldsymbol{a}(\boldsymbol{X})|\boldsymbol{\Gamma}] = \frac{P[\boldsymbol{\Gamma}|\boldsymbol{a}(\boldsymbol{X})]P[\boldsymbol{a}(\boldsymbol{X})]}{P[\boldsymbol{\Gamma}]} \quad (5)$$

We make the following assumptions:

(1) Training data are i.i.d.

(2) The *a priori* probability distribution is $P[\boldsymbol{a}(\boldsymbol{X})] \propto \exp(-\frac{1}{2} \| \hat{P}a \|^2)$.

(3) The ε-insensitive loss function is chosen as the loss function.

(4) The covariance function is $K(\boldsymbol{x}, \boldsymbol{x}')$, and $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\frac{|\boldsymbol{x}_i - \boldsymbol{x}_j|^2}{2\gamma^2})$, where $\boldsymbol{x}_i$, $\boldsymbol{x}_j$ are the input data points in $\boldsymbol{X}$.

Following Equation (6), the a posteriori probability of $\boldsymbol{a}(\boldsymbol{X})$ can be written as

$$P[\boldsymbol{a}(\boldsymbol{X})|\boldsymbol{\Gamma}] = \frac{[G(C,\varepsilon)]^N}{\sqrt{det 2\pi K_{X,X}} P[\boldsymbol{\Gamma}]} \exp\{-C \sum_{\boldsymbol{x}_i \in \boldsymbol{X}} L_\varepsilon\big(y_i - a(\boldsymbol{x}_i)\big) - \frac{1}{2}\boldsymbol{a}(\boldsymbol{X})^T K_{\boldsymbol{X},\boldsymbol{X}}^{-1} \boldsymbol{a}(X)\} \quad (6)$$

where $G(C, \varepsilon) = \frac{1}{2}\frac{C}{C\varepsilon+1}$, and $K_{\boldsymbol{X},\boldsymbol{X}} = [K(\boldsymbol{x}_i, \boldsymbol{x}_j)]$ is the covariance matrix of the data points of $\boldsymbol{X}$.

We find the maximum of Equation (7) using the so-called MAP. This requires finding the minimum of the following function

$$R_{GSVM}(a) = C \sum_{\boldsymbol{x}_i \in \boldsymbol{X}} L_\varepsilon\big(y_i - a(\boldsymbol{x}_i)\big) + \frac{1}{2}\boldsymbol{a}(\boldsymbol{X})^T K_{\boldsymbol{X},\boldsymbol{X}}^{-1} \boldsymbol{a}(X) \quad (7)$$

We can see that the risk of Gaussian SVMs is equivalent to the standard SVM. Following the discussion in Mackay (1997), Tikhonov and Arsenin (1997), Girosi (1998) and Burges (1998), we can write the solution of the minimization problem associated to Equation (8) in the following form

$$a^*(\boldsymbol{x}) = \sum_{\boldsymbol{x}_i \in \boldsymbol{X}} \beta_i K(\boldsymbol{x}_i, \boldsymbol{x}) \quad (8)$$

where $\beta_i = a_i - a_i^*$ is a combination of the Lagrange Multipliers associated to the optimization problem (Smola and Scholköpf, 2004). The a_i and a_i^* can be determined by a Quadratic Programming approach. According to Smola and Scholköpf (2004), $\forall\, i = 1, \ldots, n$, a_i and a_i^* lie in the interval $[0, C]$, and β_i consequently lies in the interval $[-C, C]$, which is the domain of the optimization problem.

There are different medium-scale and large-scale algorithms that can be used for optimizing under constraints the objective function in Equation (8). An active set algorithm which focuses on the solution of the Karush-Kuhn-Tucker (KKT) equations is used in this paper. The KKT equations are both necessary and sufficient conditions for obtaining a global solution point when the problem is a convex programming problem. Sequential Quadratic Programming (SQP) method is used to compute directly the Lagrange multipliers which balance the deviations in magnitude of the objective function and constraint gradients.

PAPER II: JIE LIU, VALERIA VITELLI, ENRICO ZIO & REDOUANE SERAOUI "A NOVEL DYNAMIC-WEIGHTED PROBABILISTIC SUPPORT VECTOR REGRESSION-BASED ENSEMBLE FOR PROGNOSTICS OF TIME SERIES DATA," *SPECIAL ISSUE OF IEEE TRANSACTIONS ON RELIABILITY*, 2014. (UNDER REVIEW)

A NOVEL DYNAMIC-WEIGHTED PROBABILISTIC SUPPORT VECTOR REGRESSION-BASED ENSEMBLE FOR PROGNOSTICS OF TIME SERIES DATA

Jie Liu, Valeria Vitelli, Enrico Zio, Senior Member, IEEE, and Redouane Seraoui

ABSTRACT

In this paper, a novel Dynamic-Weighted Probabilistic Support Vector Regression-based Ensemble (DW-PSVR-ensemble) approach is proposed for prognostics of time series data monitored on components of complex power systems. The novelty of the proposed approach consists of i) the introduction of a signal reconstruction and grouping technique suited for time series data, ii) the use of a modified Radial Basis Function (RBF) kernel for multiple time series data sets, iii) a dynamic calculation of sub-models weights for the ensemble, and iv) an aggregation method for uncertainty estimation. The dynamic weighting is introduced in the calculation of the sub-models weights for each input vector, based on Fuzzy Similarity Analysis (FSA). We consider a real case study involving 20 failure scenarios of a component of the Reactor Coolant Pump (RCP) of a typical nuclear Pressurized Water Reactor (PWR). Prediction results are given with the associated uncertainty quantification, under the assumption of a Gaussian distribution for the predicted value.

Key words: Prognostics, time series, Probabilistic support vector regression, Ensemble, Uncertainty quantification, Nuclear pressurized water reactor, Reactor coolant pump

1. INTRODUCTION

The field of research and application which aims at making use of the past and present information on the environmental, operational, and usage conditions of a component or a system in order to detect its degradation, diagnose its faults, predict, and proactively manage its failures, is called prognostics and health management [1]. This field of research articulates in various directions: fault detection and isolation, i.e. the identification and characterization of the component or system failure state, and prognostics, i.e. the prediction of the future evolution of the failure scenario. While the former is often difficult in realistic situations, the latter is even more challenging.

Various taxonomies have been proposed to categorize the different approaches for prognostics. According to [1], they can be classified into first-principle model-based approaches, reliability model-based approaches, process sensor data-driven approaches and combined approaches. It is difficult to apply first-principle model-based approaches on complex component or systems, because they are based on structural and physical assumptions, which lead in many cases to high computational costs or to excessive simplifications. Performance of reliability model-based approaches also highly depends on the underlying modeling framework which is used to derive reliable estimates. On the other hand, data-driven approaches do not make use of any physical model, relying exclusively on monitored process data from sensors related to the degradation and failure state of the component or system. Empirical techniques like Artificial Neural Networks (ANN), Support Vector Machine (SVM), Local Gaussian Regression (LGR), pattern similarity, are typical examples [1]. The advantage of data-driven approaches lies in the direct use of the measured process data for empirical learning and non-parametric estimation of component degradation and failure. Quick prediction, robustness, confidence estimation, adaptability are some desirable characteristics of these prognostic methods for practical applications.

However, data-driven approaches have some disadvantages which limit their applications. Individual models cannot always estimate the true underlying relation between heterogeneous data with the required level of accuracy, and the use of an individual model may cause over-fitting on the data set at hand. In addition, many machine learning algorithms are based on some form of local optimizer that typically shows the tendency to converge to local optima.

Combining various data-driven approaches into an ensemble is a relatively recent direction of

research, traced to improve the robustness and accuracy of the final prediction strategy. The models which compose the ensemble are called sub-models. In the original ensemble approach some strategies are proposed for building sub-models, e.g. Bayesian averaging, but more recent proposals have also been elaborated, including error-correcting output coding, Bagging, Adaboost, and boosting [2], [3], [4]. Several methods for aggregating the prediction results of sub-models have also been proposed in the literature, such as majority vote, weighted vote, Borda count, Bayes and probabilistic schemes, etc [5].

Ensemble models find application in a wide range of research fields, such as meta-modeling for the design of modern engineering systems [6], [7], discovery of regulatory motifs in bioinformatics [8], detection of traffic incidents [9], transient identification in Nuclear Power Plant (NPP) [10]. Moreover, ensembles are so flexible that they can be built using a variety of techniques (e.g., committees of neural networks [11], [12], Kalman filters [13]).

In this paper, we focus on the combination of data-driven approaches for prognostics, and more specifically on the combination of multiple Probabilistic Support Vector Regression (PSVR) sub-models [14], [15]. The case study addressed in this paper concerns the monitoring of a component in the Reactor Coolant Pump (RCP) of a NPP, with real data collected from one sensor. The time horizon for the prediction is one day and it has been fixed according to the requirements of the engineering application at hand: decisions depending on the state of the component are to be taken within this time horizon.

PSVR derives from Support Vector Regression (SVR), also called Support Vector Machine (SVM), which is one of the most popular and promising data-driven methods for prognostics. The foundations of SVM have been developed by Vapnik [16] and the method is now gaining popularity due to its remarkable generalization performance, especially when few data are available. SVM can be applied to both classification and regression problems, in the latter case being mostly called SVR. SVR and SVM have been successfully applied in many fields such as face detection, hand-writing digital character recognition, residual useful life estimation, interval forecasting of electricity price, biophysical variables estimation, and others. Some research on SVM-based ensemble models has already been carried out. Chen et al. [9] use ensemble of SVMs to detect traffic incidents. The sub-models (i.e. the models composing the ensemble) use different kernel functions and parameters to improve the classification performance. Acar and Rais-Rohami [7] treat the general weighted-sum formulation of an ensemble as an optimization problem, and then minimize an error metric to select the best

weights for the sub-models of SVM. Kurram and Kwon [17] try to achieve an optimal sparse combination of the sub-model results by jointly optimizing the separating hyperplane obtained by each SVM classifier and the corresponding weights of sub-decisions. Huang and Zhang [18] propose a new multi-feature model to construct a SVM ensemble combining multiple spectral and spatial features at both pixel and object levels. Chiang and Lee [19] propose a hybrid machine learning system by merging fuzzy multiset-based classifiers and SVM into fuzzy-SVM mixture models, which achieve consistent prediction accuracy on human protein-protein interactions. Valentini and Dietterich [20] prove that an ensemble of SVMs employing bagging of low-bias algorithms improves the generalization power of the procedure with respect to single SVM. They also present an extended experimental analysis of bias-variance decomposition in SVMs [21]. The ensemble of SVMs built with bagging and boosting greatly outperforms a single SVM in terms of classification accuracy [3].

Recently, it has been shown how to treat the SVR method as a Bayesian inference problem with Gaussian priors. The Maximum A Posteriori (MAP) solution to this problem can contextually give an estimate of the model parameters and also of the underlying function [22], [23]. Within the Bayesian treatment of SVR, an error bar for the prediction, i.e. the variance of the predicted outcome, can also be obtained [14], [15]. This Bayesian interpretation of SVR is called Probabilistic Support Vector Regression (PSVR).

An ensemble model of PSVRs is proposed in this paper with a dynamic weighting strategy. The elements of novelty of the method here proposed are various. In the previously mentioned ensembles of SVMs, all the weights were calculated during the training part and fixed for testing. However, a sub-model may perform well only on a part of the data set. Hence, the weights need to be updated considering the different data sets involved in the case study, and even different input vectors. A dynamic weighting strategy, based on Fuzzy Similarity Analysis (FSA) [24], is proposed in this paper. A dynamic weighting method is used in [25], [26], [27], to add a new classifier to the ensemble model, but weights are not adjusted to different input vectors. Moreover, in order to build an ensemble of PSVRs on different failure scenarios, a modified Radial Basis Function (RBF) is also proposed and used in this paper. In addition, a simple but efficient aggregating method is proposed to combine the outputs of the sub-models, including predicted values and associated error bars. Finally, two different strategies are proposed to form the training data sets of each sub-model on the basis of the characteristics of the data. All the novel strategies are tested in the case study concerning a component of the

RCP in a NPP.

The rest of the paper is organized as follows. Section II gives details about the proposed ensemble approach and a brief introduction to PSVR with modified RBF. Section III illustrates the case study, the available data, some necessary data pre-processing steps and how the two proposed ensemble models are constructed. Section IV presents the experimental results from the PSVR ensemble models and describes the comparison with a single PSVR model. Finally, conclusions with some considerations are drawn in Section V.

2. DYNAMIC-WEIGHTED PSVR-BASED ENSEMBLE

The strategy underlying the use of ensemble-based methods in prediction problems is to benefit from the strength of different sub-models by combining their outputs to improve the global prediction performance if compared to the result of a single sub-model. The reason why ensembles are more accurate and robust is that sub-models perform well for different data sets and/or exploration regions, and their imprecisions and one-sidedness are balanced out during the combination. Moreover, since SVM has been implemented using approximated algorithms to reduce the computational complexity, a single SVM may not always converge to the global optimum. Sometimes, the support vectors obtained from the learning are not sufficient to give good prediction performance for all unknown test examples.

One of the main advantages of PSVR is that the output is not only a point estimate, but an uncertainty estimation related to the predicted value can also be derived. The point estimation for a real application needs to be supported by a proper prediction interval, which is more informative to the NPP operator on the reliability of the estimation upon which he/she must take decisions of operation. The peculiarity of the PSVR method is exactly that it can give an error bar associated to the predicted value. After the sub-models are trained with PSVR, a simple but efficient strategy is proposed to combine their outputs. As different failure scenarios are available in the case study, the ensemble takes also advantage of the diverse information in the different scenarios for the 1-day ahead prediction output that it delivers.

In this section, we give details about the proposed Dynamic-Weighted PSVR-based Ensemble (named DW-PSVR-Ensemble in short).

2.1 Probabilistic Support Vector Regression

Depending on the choice of the loss function, we can define different Gaussian versions of

PSVR [28]. The PSVR approach proposed in the previous work [15] and used in the ongoing research makes use of the ε-insensitive Loss Function, which enables a sparse set of support vectors to be obtained [29].

2.1.1 PSVR with ε-Insensitive Loss Function

Suppose to have a time series data set $a(t)$. Partial autocorrelation analysis can help finding a time horizon H defining the best number of historical values related to the output. Hence, the input vector will be $\boldsymbol{x}(t) = (a(t-H+1), \dots, a(t))$. In regression methods, the final aim is to find a function $f^*(\boldsymbol{x})$ which estimates the function $f(\boldsymbol{x})$: $\mathbf{R}^{\mathrm{H}} \rightarrow \mathbf{R}$ describing the relation between the input data and the target: $y(t) = f\big(\boldsymbol{x}(t)\big) + \delta(t)$, ($\delta(t)$ is intrinsic noise in the data).

We briefly recall some points of the PSVR approach with ε-insensitive loss function which are critical for the clarity and consistency of DW-PSVR-Ensemble in this paper; further mathematical details on the derivation of the method can be found in [14], [15].

Let:

1) The training data $\boldsymbol{\Gamma} = \{\boldsymbol{X}, \boldsymbol{Y}\}$ be composed by independent samples $(\boldsymbol{x}(t), y(t))$, which, given $\boldsymbol{f}^*(\boldsymbol{X}) = (f^*(\boldsymbol{x}(1)), f^*(\boldsymbol{x}(2)), \dots, f^*(\boldsymbol{x}(M)))$, with M the size of training data set, are drawn from the same probability distribution. We recall that M is not equal to the size of the time series data set $a(t)$.
2) The loss function be described as follows (see [6]):

$$l(\boldsymbol{x}, \varepsilon) = \begin{cases} |\boldsymbol{x}| - \varepsilon, & |\boldsymbol{x}| \geq \varepsilon \\ 0, & |\boldsymbol{x}| < \varepsilon \end{cases}. \quad (1)$$

3) The covariance function be indicated with $\mathbf{K}_{\boldsymbol{X},\boldsymbol{X}}$, where the element $\mathbf{K}_{\mathrm{i,j}}$ of the i-th row and j-th column is $\mathrm{K}(\boldsymbol{x}(i), \boldsymbol{x}(j))$ i.e. a modified Radial Basis Function (RBF), with $\boldsymbol{x}(i), \boldsymbol{x}(j)$ being elements of $\mathbf{X}$, and $\mathrm{i,j} = 1, \dots, \mathrm{M}$.

By minimizing the following function:

$$\mathrm{R}_{\mathrm{GSVM}}(f) =$$

$$\mathrm{C} \sum_{\boldsymbol{x}(i) \in \mathbf{X}} l(y(i) - f(\boldsymbol{x}(i)), \varepsilon) + \frac{1}{2} \boldsymbol{f}(\mathbf{X})^{\mathrm{T}} \mathbf{K}_{\mathbf{X},\mathbf{X}}^{-1} \boldsymbol{f}(\mathbf{X}), \quad (2)$$

we can derive the estimator of $f(\boldsymbol{x})$ as

$$f^*(\boldsymbol{x}) = \sum_{\boldsymbol{x}(i)\in \mathrm{X}} \beta_i \mathrm{K}(\boldsymbol{x}(i), \boldsymbol{x}), \quad (3)$$

where β_i is a combination of the Lagrange Multipliers associated to the optimization problem [29] which lies in the interval $[-C, C]$. The vector $\boldsymbol{x}$ is a new input vector.

Thanks to the Bayesian approach, the estimation process leads naturally to compute the prediction variance from the predictive distribution [14]. Suppose that $\boldsymbol{x}$ is a test input vector and that $\boldsymbol{X}_M$ is the subset of all the support vectors; then, the error bar of the prediction corresponding to the test input point $\boldsymbol{x}$ is

$$\sigma^2(\boldsymbol{x}) = \sigma_\delta^2 + \sigma_t^2(\boldsymbol{x}) = \frac{2}{C^2} + \frac{\varepsilon^2(C\varepsilon+3)}{3(C\varepsilon+1)}$$

$$+ K(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{K}_{\boldsymbol{X}_M,\boldsymbol{x}}^T \boldsymbol{K}_{\boldsymbol{X}_M,\boldsymbol{X}_M}^{-1} \boldsymbol{K}_{\boldsymbol{X}_M,\boldsymbol{x}}. \quad (4)$$

2.1.2 Kernel Function and Hyperparameters

2.1.2.1 Modified Radial Basis Function Kernel

The kernel function enables to map an input vector into a higher-dimensional Hilbert space. By calculating pairwise inner products between mapped samples, kernel functions return the similarity between different samples. In fact, only kernels that fulfill Mercer's Theorem (the kernel matrix must be positive semi-definite) are valid ones and, thus, can be used in SVM [31], [32]. The most common kernel functions include the linear kernel fucntion, the polynomial kernel function and the Radial Basis Function.

In all these popular kernel functions, different inputs, i.e. different elements of $\boldsymbol{x}$(t), are treated equally in computing the inner product involved in RBF. As previously discussed, H historical values of the time series are chosen as inputs according to the partial autocorrelation analysis results. These values have, of course, different correlation structures with respect to the output. In order to reflect this difference, a modified RBF is proposed in this paper.

The traditional RBF is $K\big(\boldsymbol{x}(i), \boldsymbol{x}(j)\big) = \exp(-\frac{\|\boldsymbol{x}(i)-\boldsymbol{x}(j)\|^2}{2\gamma^2})$, and the proposed modified RBF is $K\big(\boldsymbol{x}(i), \boldsymbol{x}(j)\big) = \exp(-\frac{\langle \boldsymbol{C_a}^2, (\boldsymbol{x}(i)-\boldsymbol{x}(j))^2\rangle}{2\gamma^2})$. $\boldsymbol{C}_a = (C_1, \dots, C_H)$ is the correlation between each input and the output, in our case between different temporal lags and the output of time series data. Note that being the inputs historical values of the target, the correlation is in fact the time series autocorrelation. Suppose $\boldsymbol{A_i} = [x_i(t)], \boldsymbol{B} = [y(t)]$, with $x_i(t)$ the i-th input of $\boldsymbol{x}(t)$ and $t = 1, \dots, M$. Then, C_i is the correlation between $\boldsymbol{A_i}$ and $\boldsymbol{B}$, and so the correlation

between $x_i(t)$ and $y(t)$. As $\boldsymbol{C}_a$ is fixed vector for each sub-model, it is easy to prove that the modified RBF satisfies Mercer's Theorem. Thus, the modification of the RBF does not change the theoretical results on which the PSVR method is based.

By giving different weights to different inputs in the input vector, we can reduce the influence of the inputs less correlated with the output and make the more correlated ones more significant in the relation between the inputs and the output. Another advantage of the modified RBF is introduced in Section III, when dealing with multiple time series data.

2.1.2.2 Tuning of Hyperparameters

There are three parameters that need to be tuned (called hyperparameters hereafter), related to PSVR with ε-insensitive loss function and the modified RBF. The three hyperparameters are the penalty factor C, the sparsity control parameter ε and the width of the kernel γ.

Some methods have already been proposed in the literature to determine these hyperparameters [3], [16], [33], [34], [35]. In this paper, the interpolation method based on an innovative criterion introduced in our previous work [15] is used to obtain the best values of these three hyperparameters. Readers who are interested can refer to [15] for more details. A comparison between the proposed tuning method and a Genetic Algorithm (GA) [36] shows that the proposed method has much less computational complexity, while it gives results comparable to GA.

In this paper, all sub-models are trained using the same method; in order to keep the diversity and specificity of each sub-model, hyperparameters are tuned with respect to the performance of the individual model, instead of the global performance of the ensemble.

2.2 Ensemble-Based Approach

An ensemble-based approach is obtained by training diverse sub-models, and, then, combining their results with proper strategies. It can be proven that this can lead to superior performance with respect to a single model approach [37]. Ensemble-based approaches attempt to take advantage of each sub-model, by fusing results from all the sub-models. A simple paradigm of a typical ensemble-based approach is shown in Figure 1. Ensemble models are built on three key components: a strategy to build diverse models; a strategy to construct accurate sub-models; a strategy to combine the outputs of the sub-models in a way such that the correct decisions are amplified, while the incorrect ones are counteracted. We focus here on the latter. Proper

strategies to build diverse and accurate sub-models are described in relation to the case study.

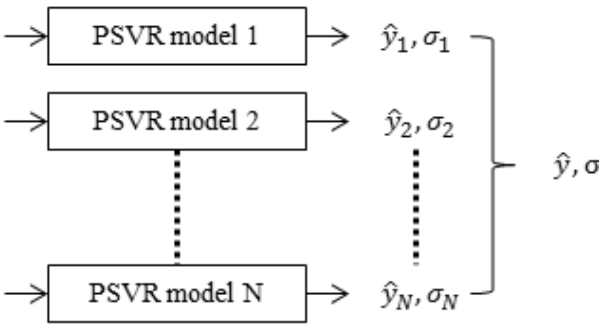

Fig. 1. Paradigm of a typical ensemble method.

In the DW-PSVR-Ensemble that we are proposing, the sub-models are built using the PSVR model presented in the previous section. The reason for not using other data-driven approaches, including other SVMs, lies on the special output structure of PSVR. The output of each sub-model built with PSVR contains a predicted value and the associated variance, assuming that the predicted value follows a Gaussian distribution.

A dynamic weighted-sum strategy is proposed to combine the outputs of the sub-models. As mentioned in Section I, different methods can be applied to calculate the weights for the sub-models. In the methods that can be found in the literature, the weights are normally fixed after the ensemble model is built. They are only updated when new sub-models are added to the ensemble or when some sub-models are changed. In some real applications with fast changing environmental and operational conditions, the performance of the ensemble model may degrade rapidly. This degradation does not always depend on the low robustness or capability to adapt of the ensemble model, but can be due to the fact that the best sub-models are not given proper weights. In this paper, a dynamic weighting strategy is thus proposed. The weights are no longer constant during the prediction, but dependent on the input vector. They are recalculated each time a new input vector arrives. Inspired by the work of [24] and considering the characteristics of PSVR, a Fuzzy Similarity Analysis (FSA) is implemented in this paper to calculate weights of different sub-models for each input vector.

2.2.1 Fuzzy Similarity Analysis

The weights of each sub-model are calculated by FSA [24].

Indeed, the performance of the models built by PSVR is highly dependent on the training data set, and more precisely on the support vectors that are derived from it. The performance of the model with respect to new incoming data is, then, highly dependent on the similarity between

the new input and the training data set. This claim is confirmed by the results in the case study. Thus, it is critical to adapt the weight assigned to the PSVR sub-model according to the similarity between the new input and the training data.

Suppose that a sub-model j out of all the N sub-models is built on the training data set $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_M)$ and the new input vector is $\boldsymbol{x}$; then, the weight of this sub-model for the new input vector can be calculated by the following steps:

1) The first step consists in calculating the Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{x}_i$for $i = 1, \dots, M,$ denoted with $\phi_i = \|\boldsymbol{x} - \boldsymbol{x}_i\|_2$.

2) The second step is to compute the point-wise similarity of $\boldsymbol{x}$ and the corresponding distance score. The distance between $\boldsymbol{x}$ and $\boldsymbol{x}_i$ is evaluated with reference to an "approximately zero" fuzzy set (FS), specified by a function that maps the Euclidean distance ϕ_i into the corresponding similarity value μ_i [38]. For the definition of the FS, triangular, trapezoidal, and bell-shaped are among the most popular functions. In the application illustrated in this work, the following bell-shaped function is used:

$$\mu_i = \exp(-(-\ln(\alpha)/\beta^2){\phi_i}^2). \quad (5)$$

The arbitrary parameters α and β in (5) can be fixed by the analyst: the larger the value of the ratio -ln(α)/β^2, the narrower the fuzzy set and the stronger the definition of similarity [24].Then, the distance score $d_i = 1 - \mu_i$ is computed.

3) The third step is to find the minimum d^j, associated to the j-th sub-model, of the d_is, for $i = 1, \dots, M$. Then, $\widetilde{\omega}_j = (1 - d^j)\exp(-\frac{d^j}{\beta})$, is the strategy to obtain the weight of sub-model j.

By repeating the previous steps for all sub-models j, $j = 1, \dots, N$, we obtain all the weights $\widetilde{\omega}_j$. The final weight of each sub-model for the new input vector $\boldsymbol{x}$ can be calculated as $\omega_j = \widetilde{\omega}_j / \sum_{k=1}^{N} \widetilde{\omega}_k$. These final weights can be applied in the ensemble paradigm to derive the output of the ensemble model for $\boldsymbol{x}$, as explained in the next step below.

2.2.2 Combining sub-models outputs

Figure 2 shows the paradigm of DW-PSVR-Ensemble, where N is the number of sub-models, $\boldsymbol{x}(t)$ is a new input vector arriving at time t, $w_j(t)$ is the weight assigned to the j-th sub-

model for the new input vector, $\hat{y}_j(t)$ and $\sigma_j{}^2(t)$ are the predicted value and corresponding variance for the j-th sub-model given by (3) and (4), and $\hat{y}(t)$ and $\sigma^2(t)$ are the final outputs of the ensemble model.

TABLE I

Characteristics of Raw and Reconstructed Scenarios

Scenario	Size of Raw Data	Best Number of Historical values H	Size of Reconstructed Data
1	2277	7	2265
2	385	3	373
3	385	3	373
4	2027	14	2015
5	2027	8	2015
6	2027	8	2015
7	1391	13	1379
8	1391	4	1379
9	1391	4	1379
10	1391	4	1379
11	3124	12	3112
12	562	7	550
13	562	9	550
14	562	9	550
15	964	2	952
16	2767	8	2755
17	2767	7	2755
18	1061	7	1049
19	1061	12	1049
20	861	9	849

The output of each PSVR-based sub-model is a Gaussian distribution. The proposed simple but efficient strategy for combining sub-models results is by taking a weighted-sum of Gaussian distributions, which means that $N(\hat{y}(t), \sigma^2(t)) = \sum_{j=1}^{N} \omega_j(t) N(\hat{y}_j(t), \sigma_j{}^2(t))$, with $N(\hat{y}(t), \sigma^2(t))$ denoting a Gaussian distribution with mean value $\hat{y}(t)$ and variance $\sigma^2(t)$.

From this, we can derive the fact that $\hat{y}(t) = \sum_{j=1}^{N} \omega_j(t)\hat{y}_j(t)$ and $\sigma(t) = \sqrt{\sum_{j=1}^{N} \omega_j{}^2(t)\sigma_j{}^2(t)}$, if we assume sub-models results to be uncorrelated.

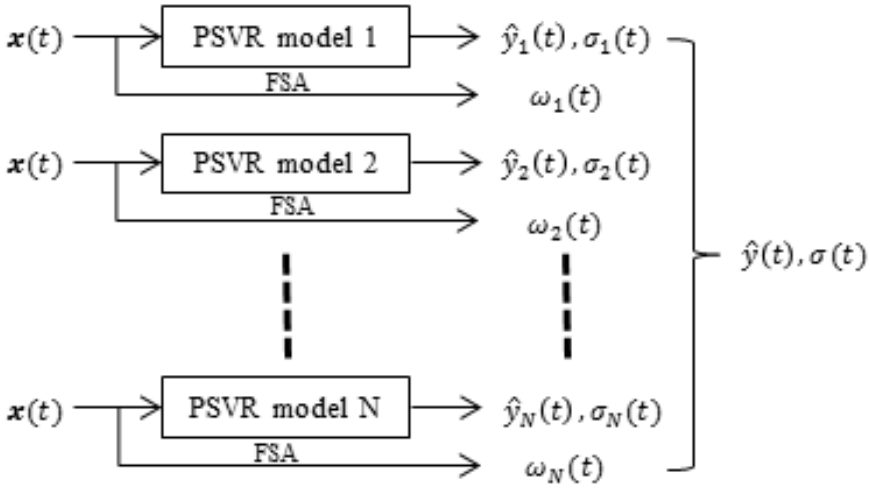

Fig. 2. Paradigm of the proposed DW-PSVR-Ensemble.

Note that all the sub-models weights and outputs are a function of t, which means that they are all dependent on the input vector of the ensemble model.

3. CASE STUDY DESCRIPTION

The real case study considered in this paper concerns the 1-day ahead prediction of leak flow of the first seal of the RCP of a NPP. It is important to predict the leak flow of the component in a near coming future, as the RCP is a critical system to maintain the safety of NPP. According to the company needs, the purpose is 1-day ahead prediction.

In this section we describe the time series data and briefly recall the data pre-processing steps. We detail the strategies to build accurate and diverse sub-models.

3.1 Data Description and Pre-processing

In the data provided by EDF, there are totally 20 failure scenarios concerning the leak flow from 10 different NPPs. They are named Scenario 1, Scenario 2, …, Scenario 20 in the following sections of the paper. These data are monitored every four hours. As these data are time dependent and recorded within different time windows, only failure scenarios coming from the same NPP have the same size. From the second column of Table I, we can see that the size of the failure scenarios can vary from 385 to 3124 data points. In some of the scenarios, there are missing data points and outliers.

The first step is to delete the outliers and reconstruct the missing data points. This latter task has been accomplished using a local polynomial regression fitting. All details can be found in [15].

Figure 3 shows the data of Scenario 1 before and after the pre-processing performed in this step of the analysis.

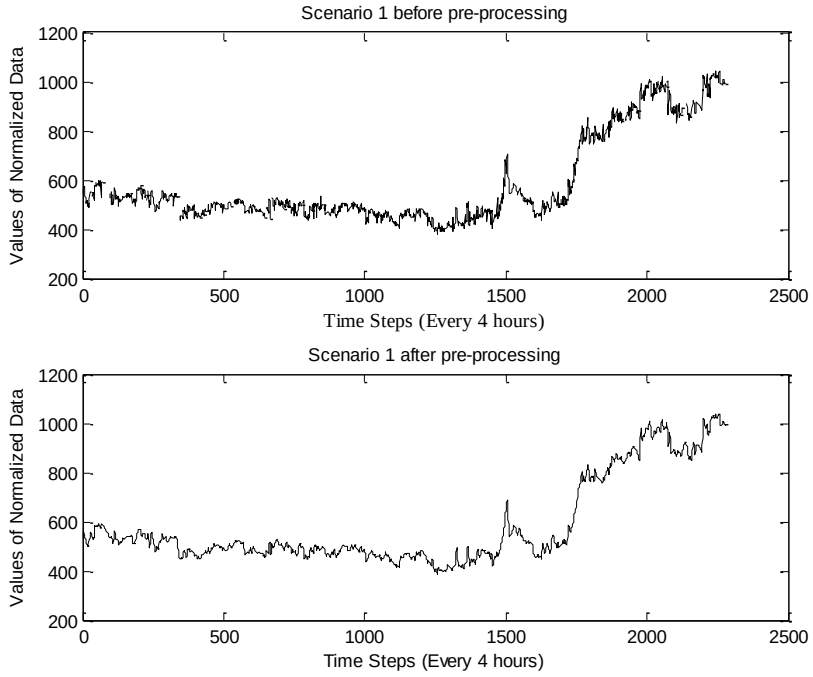

Fig. 3 Data of Scenarios 1 before and after pre-processing.(the upper one is the raw data and the lower one is the data after pre-processing)

All the data points of all failure scenarios are, then, normalized from 0 to 1.

3.2 Strategies to build Sub-models

Since we have a time series data set and since there is no other information available related to the target except for a set of monitored data directly related to the condition of the component of interest, the inputs of the model can only be a set of historical values. Before building the sub-models of the ensemble, we, thus, need to decide the best number of historical values to be used as inputs.

3.2.1 Sub-Model Identification

Suppose $a(t)$ represents an instance of the time series data of one failure scenario. For 1-day ahead prediction, the output $y(t)$ is $a(t+6)$, because the signals are monitored every four hours. In order to decide the best H for selecting the input vector $\boldsymbol{x}(t) = (a(t-H+$

$1), \dots, a(t))$ most related to the output, a partial autocorrelation analysis is carried out on each failure scenario, i.e. correlation between the output and different temporal lags is computed. Figure 4 shows the results of this analysis on Scenario 1, where the x and y axis represent, respectively, the temporal lag (a multiple of four hours) and the corresponding empirical partial autocorrelation. The bounds of a 95% confidence interval are also shown with the dashed lines in the figure. The correlation decreases with the lag (although not linearly), and after a lag of seven time steps, for Scenario 1 it is no longer comparable with the values observed for lags smaller than 7, i.e. the best choice is $H_1 = 7$.

A best value H_i is, thus, found for Scenario i, for $i = 1, 2, \dots, 20$; but this value is not the same for all scenarios, as shown in the third column of Table I. When building an ensemble model, however, a unified size of input vector would simplify the model, since a single value of H is applied for all scenarios to reconstruct the data. If we choose a small H, some useful information would be ignored for those scenarios with larger best H; in contrast, choosing a large H would bring some perturbations to scenarios with smaller best H. In order to solve this problem, we propose the modified RBF, where $\boldsymbol{C}_a$, calculated by partial autocorrelation analysis, controls the contribution of each variable of the input vector, when H is chosen as the largest of all the failure scenarios. For one scenario with smaller best H_i, the values for the last $H - H_i$ elements of the vector $\boldsymbol{C}_a$ are very small compared to the first H_i elements, because their correlations with the output are very weak. In this case study, we choose the biggest time step of all the scenarios, i.e. $H = 14$.

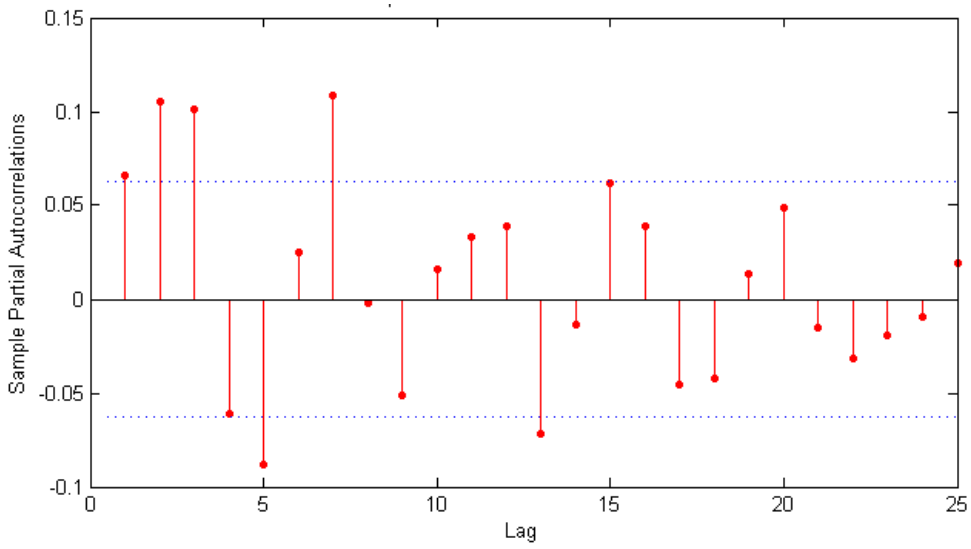

Fig. 4 Partial autocorrelation function of Scenario 1 with respect to time lags (multiples of four hours). Dotted lines are bounds of a 95% confidence interval.

The two parameters in FSA, α and β are set to 0.005 and 0.5 by trial and error.

3.2.2 Two Strategies to Build Sub-Models

Bagging and boosting are two of the most popular strategies to build diverse sub-models of an ensemble. However these methods are more suitable with scarce data. In our case, there are enough data (20 failure scenarios), so that two simple but efficient and reasonable strategies can be proposed.

Thanks to the sub-model identification process described before, the data for each failure scenario has been reconstructed with same structure, where the input vector is $\boldsymbol{x}(t) = (a(t-13), \dots, a(t))$, and the corresponding output is $y(t) = a(t+6)$, and t takes every possible values in each scenario. The size of each failure scenario after reconstruction is listed in the fourth column of Table I.

With multiple failure scenarios available, the simplest and most immediate strategy is to build a sub-model on each failure scenario, so that the number of sub-models equals the number of failure scenarios. Because of the frequently changing operational and environmental conditions in NPP, each scenario can represent a specific process, and thus sub-models built in such a way show enough diversity between each other. Another simple but effective strategy is to mix all the data points from all failure scenarios, and then divide them into different groups according to their target values $y(t)$. A sub-model is then trained on each group. This strategy is inspired by the intrinsic structure of SVM/PSVR. Performance of SVM depends highly, although not only, on the training data set (or support vectors). Sub-models built on training data set considering different ranges of output values can strengthen the specialty of each sub-model on particular characteristics of the input vectors. This strategy can make the sub-models perform well on different text examples but worse on others. The proposed weighted-sum strategy to combine the outputs of sub-models will be shown to outperform the individual model. These two strategies are named Ensemble 1 and Ensemble 2, for convenience.

3.2.3 Comparison of DW-PSVR-Ensemble with Single PSVR

The ensemble model is expected to give better results than a single PSVR model. To verify this claim, a comparison between a single PSVR model and the proposed DW-PSVR-Ensemble is carried out on the considered case study.

Each time one out of 20 failure scenarios is chosen as the test data set (named Observed

Scenario), the other 19 failure scenarios (named Reference Scenarios) are used to construct the ensemble model with the two previously proposed strategies. A PSVR model is also trained on the Observed Scenario for comparison (it is named Single PSVR to be distinguished from the two ensemble models). The size of the training data set for all PSVR models is fixed at 200 for the fairness of comparison. The choice of the size is decided by trial and error in order not to increase too much the computational complexity in time and storage, which increases exponentially with the size of the training data set, and in order to guarantee the accuracy of the model.

The steps of comparison are the following:

1) Choose the training data set for Ensemble 1: 200 data points equidistantly distributed for each Reference Scenario are selected. Totally, 19 sub-models can be trained with PSVR, each trained on 200 data points from each scenario.

2) Choose the training data set for Ensemble 2: the normalized data of 19 Reference Scenarios are sorted according to the output value of each data point and then divided into 10 groups, with the output value in the intervals of [0, 0.1], [0.1, 0.2], …, [0.9, 1]. For each group, if the size is bigger than 200, 200 data points equidistantly distributed in the group are chosen, if not, all the points in the group are used in the training data set. For the first eight groups, the size of training data set is 200, while for the last 2, the training data sets contain only 90 and 33 data points. Ten sub-models are built with PSVR on these training data sets.

3) Choose the training data set for the single PSVR: the first 200 data points of the Observed Scenario are chosen to train one single PSVR model for regression on it.

4) Calculation of Mean Absolute Error (MAE), Mean Relative Error (MRE), width of prediction intervals with 95% confidence level (PI_Width), and Coverage percentage of prediction intervals with 95% confidence level (PI_Coverage) of the outputs of Ensemble 1, Ensemble 2 and Single PSVR.

5) Comparison of Ensemble 1, Ensemble 2 and Single PSVR considering prediction accuracy, uncertainty of estimation, robustness, speed of the prediction, and adaptability.

The results and comparisons between these three models are presented in the next section.

4. RESULTS

In this section, the results from Ensemble 1, Ensemble 2 and Single PSVR are compared with respect to different aspects.

4.1 Prediction Accuracy and Uncertainty Estimation

Figures 5, 6, and 7 are the prediction results (including prediction values $\hat{y}$ and prediction interval with 95% confidence level, i.e. $[\hat{y} - 1.97\sigma, \hat{y} + 1.97\sigma]$, where σ is the variance of the assumed Gaussian distribution of the predicted value) of Scenario18, respectively from Ensemble 1, Ensemble 2 and Single PSVR. It is clear that Single PSVR cannot follow the development of Scenario 18. There is no such problem with Ensemble 1 and Ensemble 2, because the training data set contains more information than that of Single PSVR. Moreover, if the target values are higher than 0.8, Ensemble 1 gives better results than Ensemble 2, with more stable prediction intervals. This is caused by the scarceness of the training data set for the last two sub-models of Ensemble 2, which are supposed to be experts on the prediction of the data points with output values in the intervals of [0.8, 0.9] and [0.9, 1.0].

We cannot prove the superiority of the ensemble compared to Single PSVR model only by inspection of the prediction results of Scenario 18. Table II shows the prediction results for all the failure scenarios, considering MAE, MRE, PI_Width, and PI_Coverage. Figures 8, 9, 10 and 11 are the boxplots of these results to illustrate the differences between these three models. We notice that Single PSVR can give comparable prediction accuracy to ensemble models for some failure scenarios, but not for all of them. The bad results of Single PSVR are caused by the fact that the prediction results are highly dependent on the training data set. Moreover, the hyperparameters optimization is also critical to the performance of PSVR. Well-chosen hyperparameters values can improve the performance of PSVR. However, the optimization method can easily converge to a local extreme, which results into a good performance at the beginning but very bad at the end of the scenario.

These unstable results from Single PSVR prove the necessity of the ensemble approach for avoiding the limits of Single PSVR in attaining the desired accuracy and robustness of the model. The prediction results from Ensemble 1 and Ensemble 2 confirm the practicability and efficiency of the DW-PSVR-Ensemble approach. We should note that the performance of the DW-PSVR-Ensemble is influenced by the grouping strategy for deciding the training data set for each sub-model.

The calculation of the dynamic weights of each sub-model for each data point brings additional computational burden to the prediction with ensemble, but it is acceptable in real applications with large data set. Support the size of the whole training dataset is N and the number of input vector is M, the computation complexity for a fixed-weighted PSVR-based ensemble and Single-PSVR is both $O(NM)$, and that for DW-PSVR-Ensemble is $O(2NM)$.

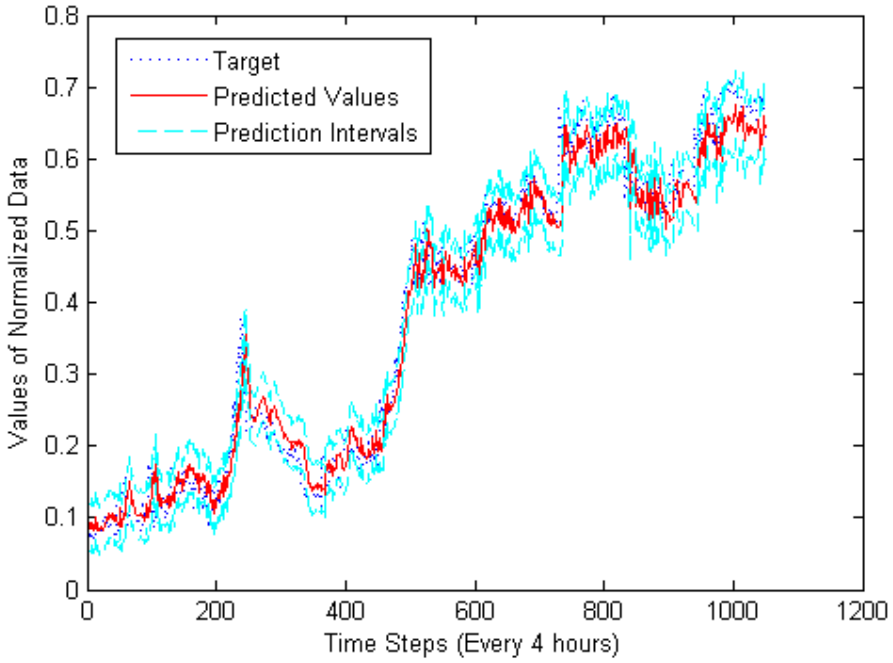

Fig. 5 Prediction results of Ensemble 1 for Scenario 18.

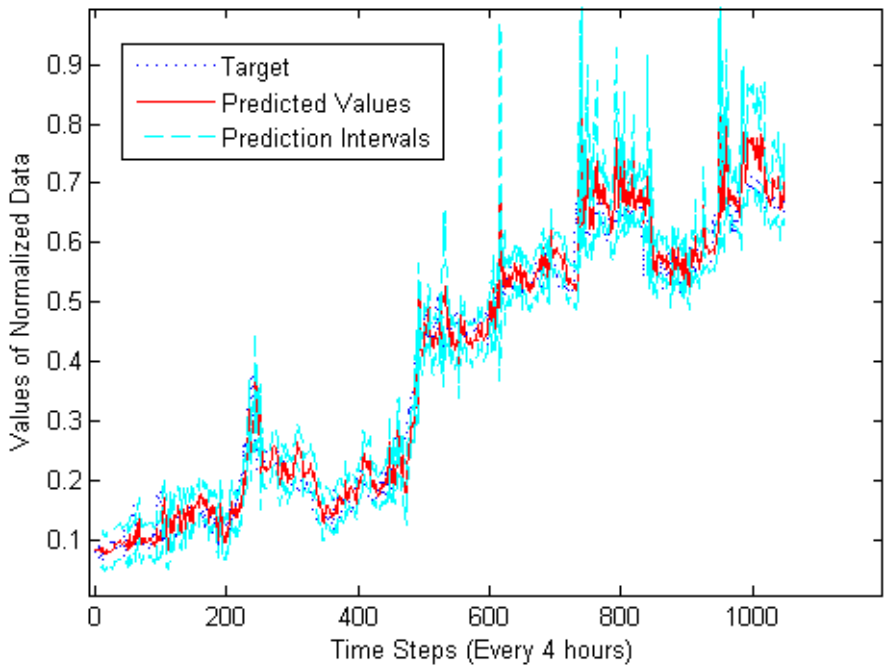

Fig. 6 Prediction results of Ensemble 2 for Scenario 18.

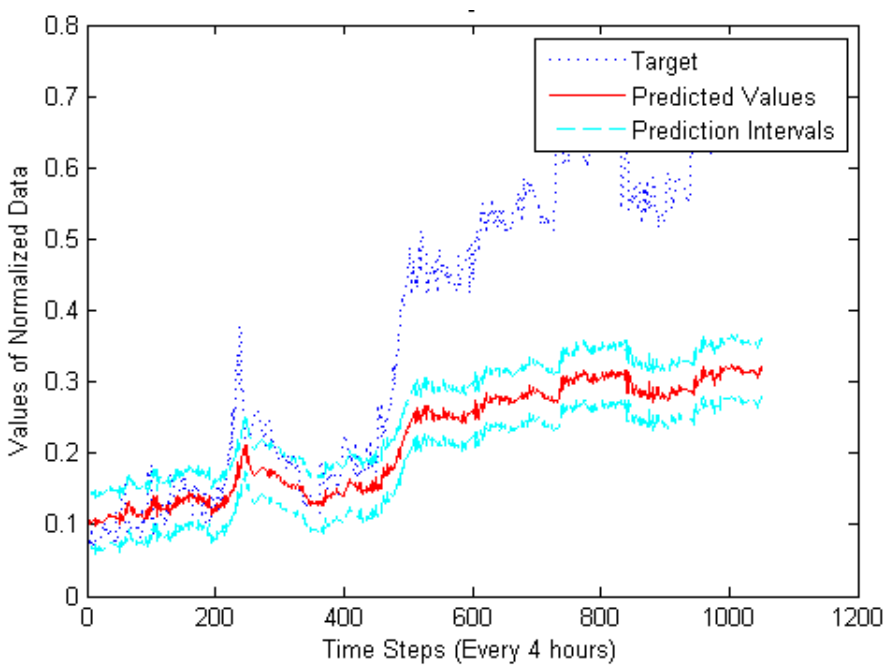

Fig. 7 Prediction results of Single PSVR for Scenario 18.

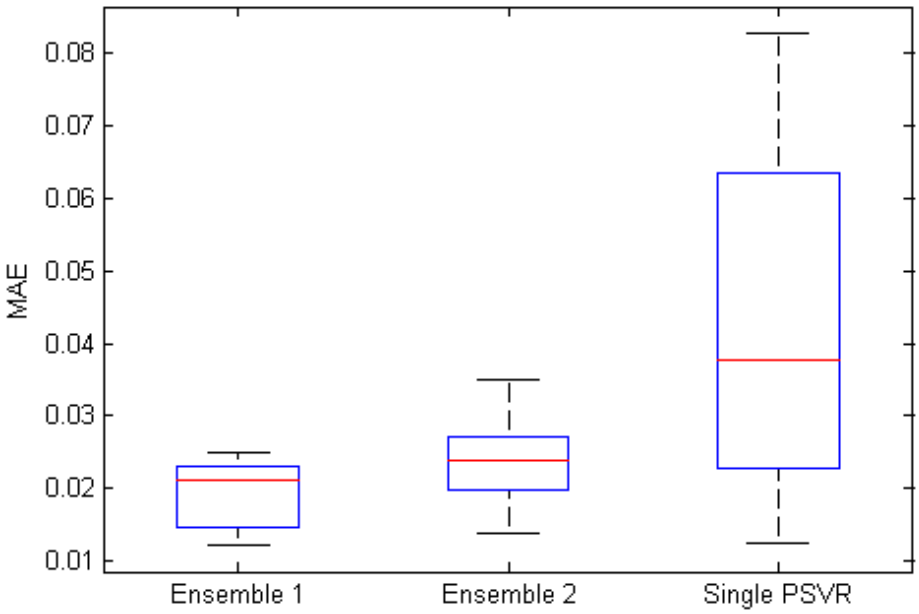

Fig. 8 MAE of prediction results of Ensemble 1, Ensemble 2 and Single PSVR, for all 20 failure scenarios.

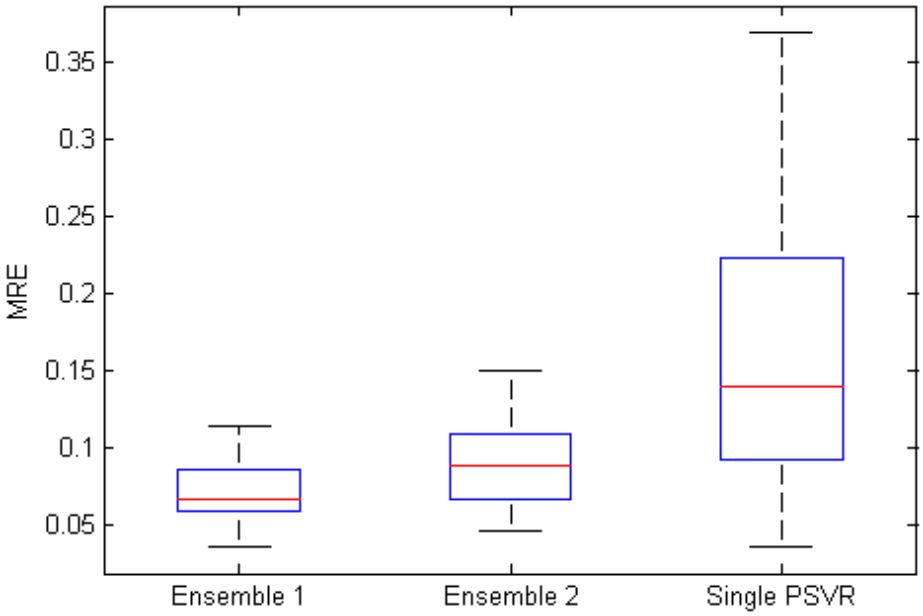

Fig. 9 MRE of prediction results of Ensemble 1, Ensemble 2 and Single PSVR, for all 20 failure scenarios.

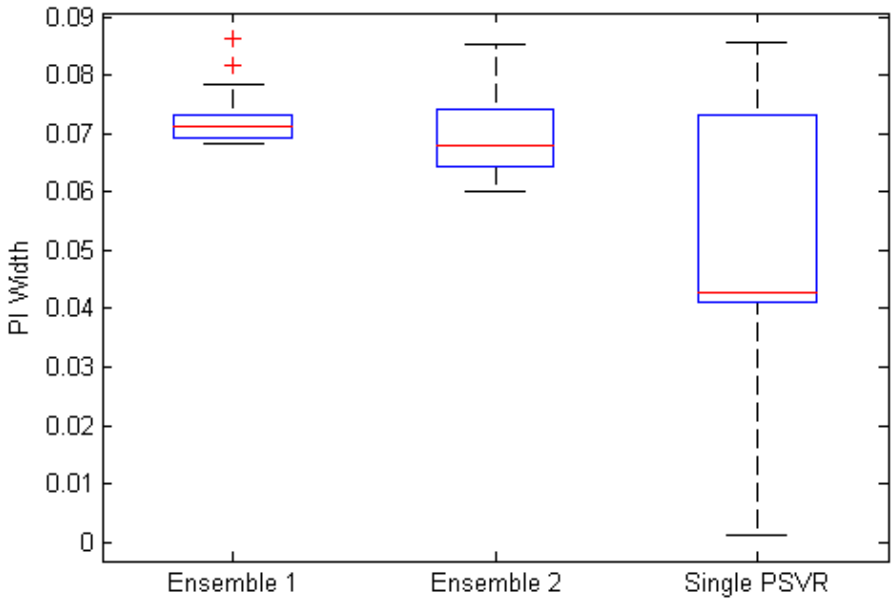

Fig. 10 Width of Prediction intervals with 95% confidence level of prediction results of Ensemble 1, Ensemble 2 and Single PSVR, for all 20 failure scenarios.

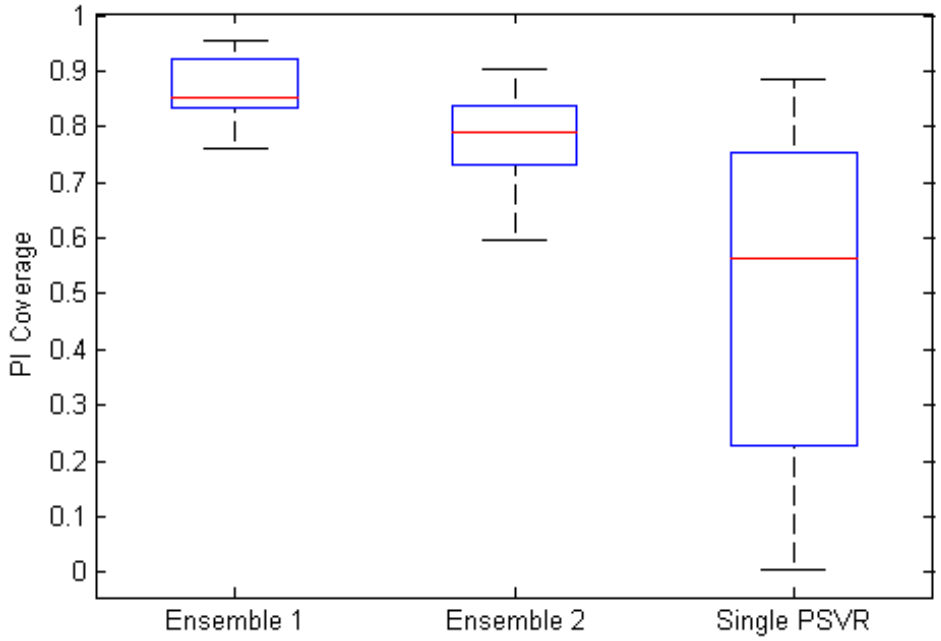

Fig. 11 Coverage of Prediction intervals with 95% confidence level of prediction results of Ensemble 1, Ensemble 2 and Single PSVR, for all 20 failure scenarios.

4.1 Robustness

From Figures 8, 9, 10, and 11, it is seen that ensemble models give more stable prediction results compared to Single PSVR model. Single PSVR model cannot properly handle the noise in the data and it is difficult to find the global optimal values of hyperparmeters, even with the modified RBF proposed in this paper. The weighted-sum ensemble models can decrease the influence of the noise by combining the prediction outputs of the sub-models; this is one reason for which ensemble models can give stable results, i.e. the ensemble models are more robust

compared to Single PSVR.

TABLE II

Comparison of Prediction Results of Ensemble 1, Ensemble 2 and Single PSVR, for all 20 Scenarios

	MAE			MRE			PI_Width			PI _Coverage		
Scenario	Ensemble 1	Ensemble 2	Single PSVR	Ensemble 1	Ensemble 2	Single PSVR	Ensemble 1	Ensemble 2	Single PSVR	Ensemble 1	Ensemble 2	Single PSVR
1	0.0172	0.0239	0.0633	0.0864	0.1128	0.3704	0.0708	0.0675	0.0131	0.8962	0.7660	0.0318
2	0.0125	0.0165	0.0124	0.0355	0.0468	0.0360	0.0725	0.0601	0.0422	0.9491	0.8606	0.8740
3	0.0229	0.0243	0.0249	0.0442	0.0476	0.0474	0.0864	0.0853	0.0431	0.8767	0.8686	0.6595
4	0.0232	0.0301	0.0687	0.0622	0.0806	0.1527	0.0730	0.0786	0.0414	0.8314	0.7382	0.4356
5	0.0193	0.0238	0.0207	0.0580	0.0730	0.0616	0.0697	0.0678	0.0452	0.8437	0.7843	0.7126
6	0.0218	0.0228	0.0314	0.0643	0.0701	0.0909	0.0727	0.0681	0.0688	0.8418	0.8109	0.6518
7	0.0141	0.0196	0.0245	0.0594	0.0867	0.0932	0.0722	0.0649	0.0482	0.9369	0.8347	0.5946
8	0.0136	0.0199	0.0227	0.0845	0.1228	0.1261	0.0695	0.0623	0.0731	0.9398	0.7962	0.8209
9	0.0156	0.0237	0.0706	0.0682	0.1067	0.3179	0.0694	0.0635	0.0731	0.9101	0.7252	0.3416
10	0.0121	0.0178	0.0405	0.0734	0.1062	0.1756	0.0694	0.0615	0.0412	0.9565	0.8419	0.4336
11	0.0231	0.0350	0.0229	0.0893	0.1507	0.0910	0.0695	0.0706	0.0783	0.8143	0.5964	0.8515
12	0.0235	0.0273	0.0828	0.0664	0.0915	0.2343	0.0783	0.0755	0.0014	0.8491	0.7291	0.0036
13	0.0141	0.0139	0.0210	0.1139	0.1109	0.1245	0.0682	0.0611	0.0411	0.9364	0.9055	0.7945
14	0.0234	0.0178	0.0661	0.0562	0.0525	0.2025	0.0818	0.0714	0.0411	0.8636	0.8800	0.1964
15	0.0222	0.0222	0.0556	0.0680	0.0684	0.1796	0.0687	0.0667	0.0011	0.8298	0.8193	0.0032
16	0.0216	0.0254	0.0352	0.0649	0.0650	0.2115	0.0693	0.0674	0.0731	0.8316	0.7514	0.6744
17	0.0203	0.0287	0.0223	0.1024	0.1452	0.1094	0.0692	0.0707	0.0855	0.8334	0.6958	0.8868
18	0.0249	0.0309	0.0634	0.0892	0.1048	0.2520	0.0717	0.0807	0.0086	0.7617	0.7283	0.0391
19	0.0208	0.0272	0.0536	0.0869	0.1085	0.2566	0.0733	0.0760	0.0412	0.8532	0.7731	0.2593
20	0.0225	0.0250	0.0421	0.0559	0.0624	0.0986	0.0736	0.0731	0.0732	0.8481	0.8080	0.5324
Average	0.0194	0.0238	0.0422	0.0715	0.0907	0.01616	0.0725	0.0696	0.0467	0.08702	0.7857	0.4899

4.3 Adaptability

Once a model is trained with PSVR, the hyperparameters values are specified for the training data set. The capability of the model is limited by the values of the hyperparmameters and the training data set, so the model adaptability is reduced. In general, the parameters and training data set need to be updated before the model can adapt to a new scenario [39]. With ensemble models, even if the new scenario is not similar to the training data sets of all the sub-models, the ensemble model can still give satisfactory results. This is a benefit of the ensemble-based approach. When the ensemble model cannot deal with the new data points or new scenario, there is no need to re-train the whole ensemble: simply one can add a sub-model or update some of the sub-models for adapting to the context changing, dynamic environment [40], [41, [42].

To update the Ensemble 1 with new data points, we can only add new sub-models, because the sub-models are trained on different plant scenarios. By updating the existing sub-models, we risk losing their integrity and completeness on the specified scenarios. For Ensemble 2, we can add new sub-models or update some of the existing sub-models without losing the capacity of the "old" ensemble. For example, when new data points are available at the range of [0.8, 0.9] or [0.9, 1], they can be used to update the corresponding sub-models, as for now there are not enough training data set for the last two sub-models of Ensemble 2. When there are new data which exceed the current range [0, 1], a new sub-model trained on these data can be added to the ensemble models.

In conclusion, the trained ensemble model adaptability is much stronger than the one of a single PSVR model.

5. CONCLUSIONS

In this paper, we propose an innovative dynamic-weighted PSVR-based ensemble approach for short-term prediction (1-day ahead prediction). Fuzzy similarity analysis is integrated to calculate the specific weights of the sub-models of the ensemble for each new input vector without bringing too much computational burden. A modified RBF kernel is used to discriminate the different correlation of the different inputs with the output.

According to the characteristics of the available time series data in the case study, two strategies are proposed to form an ensemble model: one considering different scenarios and the other selecting different ranges of output values. In both cases, the proposed ensemble approach performs well in the real case study of signals recorded on a NPP component. Compared to

single model PSVR, the proposed ensemble models outperform on prediction accuracy, robustness and adaptability.

Further research needs to be carried out, for optimizing the numbers of sub-models and for obtaining a more careful tuning of hyperparameters.

REFERENCES

E. Zio, "Diagnostics and Prognostics of Engineering Systems: Methods and Techniques," in *Diagnostics and Prognostics of Engineering Systems: Methods and Techniques,* ch. *17,* Engineering Science Reference, 2012, USA, pp. 333-356.

T. G. Dietterich, "Ensemble methods in machine learning," in *Proc. MCS '00 Proceedings of the First International Workshop on Multiple Classifier Systems,* 2000, pp. 1-15.

H. C. Kim, S. Pang, H. M. Je, D. Kim, and S. Y. Bang, "Constructing support vector machine ensemble," *Pattern Recogn.,* vol. 36, pp. 2757-2767, 2003.

C. Hu, B. D. Youn, P. Wang and J. T. Yoon, "Ensemble of data-driven prognostic algorithms for robust prediction of remaining useful life," *Reliab. Eng. Syst. Saf.,* vol. 103, no. 2012, pp. 120-135, 2012.

R. Polikar, "Ensemble based system in decision making," *IEEE Trans. Circuits Syst. Mag.,* vol. 6, no. 3, pp. 21-45, 2006.

T. Goel, R. T. Raftka, W. Shyy, and N. V. Quepo, "Ensemble of surrogates," *Struct. Multidiscip. O.,* vol. 33, no. 3, pp. 199-216, 2007.

E. Acar, M. Rais-Rohani, "Ensemble of metamodels with optimized weight factors," *Struct. Multidiscip. O.,* vol. 37, no. 3, pp. 279-294, 2009.

J. Hu, Y. D. Yang, Kihara and D. Kihara, "EMD: an ensemble algorithm for discovering regulatory motifs in DNA sequences," *BMC bioinformatics,* vol. 7, pp. 342, 2006.

S. Chen, W. Wang, and H. Zuylen, "Construct support vector machine ensemble to detect traffic incident," *Expert Syst. Appl.,* vol. 36, no. 8, pp. 10976-10986, 2009.

P. Baraldi, R. Razavi-Far, and E. Zio, "Classifier-ensemble incremental learning procedure for nuclear transient identification at different operational conditions," *Reliab. Eng. Syst. Saf.,* vol. 95, no. 7, pp. 480-488, 2010.

M. P. Perrone, and L. N. Cooper, "When networks disagree: ensemble methods for hybrid neural networks" In *proc. Neural Networks for Speech and Image,* Chapman-Hall, 1993.

C. M. Bishop, "Neural networks for pattern recognition," *Oxford University Press,* UK, 2005.

G. Evensen, "The ensemble Kalman filter: theoretical formulation and practical implementation," *Oc. Dyn.,* vol. 53, no. 4, pp. 343-367, 2003.

J. B. Gao, S. R. Gunn, C. J. Harriset and M. Brown, "A Probabilistic Framework for SVM Regression and Error Bar Estimation," *Mach. Learn.,* vol. 46, no. 1-3, pp. 71-89, 2002.

J. Liu, R. Seraoui, V. Vitelli and E. Zio, "Nuclear Power Plant Components Condition Monitoring by Probabilistic Support Vector Machine," *Ann. Nucl. Energy,* vo. 56, pp. 23-33, 2013.

V. N. Vapnik, "The Nature of Statistical Learning Theory," *Springer,* New York, 1995.

P. Gurram and H. Kwon, "Sparse kernel-based ensemble learning with fully optimized kernel parameters for hyperspectral classification problems," *IEEE Trans. Geosci. Remote,* vol. 51, no. 2, pp. 787-802, 2013.

X. Huang and L. Zhang, "An SVM ensemble approach combining spectral, structural, and semantic features for the classification of High-resolution remotely sensed imagery," *IEEE Trans. Geosci. Remote,* vol. 51, no. 1, pp. 257-272, 2013.

J. H. Chiang and T. L. M. Lee, "In silico prediction of human protein interactions using fuzzy-SVM mixture models and its applications to cancer research," *IEEE Trans. Fuzzy Syst.,* vol. 16, no. 4, pp. 1087-1095, 2008.

G. Valentini and T. G. Dietterich, "Low bias bagged support vector machines," *in Proc. the 20th international conference on machine learning (ICML-2003),* USA: Washington DC, 2003.

G. Valentini and T.G. Dietterich, "Bias-Variance analysis of support vector machines for the development of SVM-based ensemble methods," *J. Mach. Learn. Res.,* vol. 5, pp. 725-775, 2004.

D. S. Kim, J. H. Kim, M. Gyunna and J. W. Kim, "Uncertainty analysis of data-based models for estimating

collapse moments of wall-thinned pipe bends and elbows," *Nucl. Eng. & Tech.,* vol. 44, no. 3, pp. 323-330, 2012.

P. Sollich, "Probabilistic interpretations and Bayesian methods for support vector machines," King's College London, Tech. Rep., London, UK, 1999.

E. Zio and F. Di Maio, "A data-driven fuzzy approach for predicting the remaining useful life in dynamic failure scenarios of a nuclear system," *Reliab. Eng. Syst. Saf.,* vo. 95, pp. 45–57, 2010.

M. D. Muhlbaier, A. Topalis and R. Polikar, "Learn++.NC: combining ensemble of classifiers with dynamically weighted consult-and-vote for efficient incremental learning of new classes," *IEEE Trans. Neural Networ.,* vol. 20, no. 1, pp. 152-168, 2009.

X. Yang, B. Yuan and W. Liu, "Dynamic weighting ensembles for incremental learning," *Pattern Recognition, 2009. CCPR 2009. Chinese Conference on,* China, Nov. 2009.

R. Razavi-Far, P. Baraldi and E. Zio, "Dynamic weighing ensembles for incremental learning and diagnosing new concept class faults in nuclear power systems," *IEEE Trans. Nucl. Sci.,* vol. 59, no. 5, pp. 2520-2530, Oct. 2012.

W. Chu, S. S. Keerthi and C. J. Ong, "Bayesian Support Vector Regression Using a Unified Loss Function," *IEEE Trans. Neural Networ.,* vol. 15, no. 1, pp. 29-44, 2004.

A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Stat. Comp.,* vo. 14, no. 3, pp. 199-222, 2004.

D. J. MacKay, "Gaussian processes, a replacement for neural networks," NIPS tutorial 1997, Cambridge University, 1997.

H. Minh, P. Niyogi and Y. Yao, "Mercer's theorem, feature maps, and smoothing," *Lect. Notes Comp. Sci.,* vol. 4005, pp. 154-168, 2006.

A. Belghith, C. Collet and J. P. Armspach, "Change Detection based on a support vector data description that treats dependency," *Pattern Recogn.,* vol. 34, pp. 275-282, 2013.

D. J. MacKay, "Bayesian Modelling and Neural Networks," Ph.D. dissertation, California Institute of Technology, Pasadena, CA, 1991.

H. Akaike, "A new look at the statistical model identification," *IEEE Trans. Auto. Control.,* vol. 19, no. 6, pp.716–723, 1974.

N. Murata, S. Yoshizawa and S. Amari, "Network information criterion–determining the number of hidden units for artificial neural network models," *IEEE Trans. Netw.,* vo. 5, pp.865–872, 1994.

R. Ak, Y.F. Li, V. Vitelli and E. Zio, "Adequacy Assessment of a Wind-integrated Power System using Neural Network - based Interval Predictions of Wind Power Generation and Load," *IEEE Trans. Power Syst.,* to be published.

E. Bauer and R. Kohavi, "An empirical comparison of voting classification algorithms: Bagging, boosting, and variants," *Mach. Learn.,* vol. 36, no. 1-2, pp. 105-139, 1999.

A. Joentgen, L. Mikenina, R. Weber and H.J. Zimmermann, "Dynamic fuzzy data analysis based on similarity between functions," *Fuzzy Set. Syst.,* vol. 105, no. 1, pp. 91-90, 1999.

J. Kivinen, A.J. Smola and R.C. Willliamson, "Online learning with kernels," *IEEE Trans. Signal proces.,* vol. 52, no. 8, pp. 2165-2176, 2004.

R. Polikar and C. Alippi, "Guest Editorial Learning in Nonstationary and Evolving Environments," *IEEE Trans. Neural Networ. & Learn. Syst.,* vol. 25, no. 1, pp. 9-11, 2014.

J. B. Gomes, M. M. Gaber, P. A. C. Sousa and E. Menasalvas, "Mining recurring concepts in a dynamic feature space," *IEEE Trans. Neural Networ. & Learn. Syst.,* vol. 25, no. 1, pp. 95-110, 2014.

D. Brzezinski and J. Stefanowski, "Reacting to different types of concept drift: the accuracy updated ensemble algorithm," *IEEE Trans. Neural Networ. & Learn. Syst.,* vol. 25, no. 1, pp. 81-94, 2014.

PAPER III: JIE LIU, VALERIA VITELLI, ENRICO ZIO & REDOUANE SERAOUI "A DYNAMIC WEIGHTED RBF-BASED ENSEMBLE FOR PROGNOSTICS OF NUCLEAR COMPONENTS," *INTERNATIONAL JOURNAL OF PROGNOSTICS AND HEALTH MANAGEMENT*, 2014. (UNDER REVIEW)

A DYNAMIC WEIGHTED RBF-BASED ENSEMBLE FOR PROGNOSTICS OF NUCLEAR COMPONENTS

Jie Liu, Valeria Vitelli, Enrico Zio and Redouane Seraoui

ABSTRACT

In this paper, an ensemble approach is proposed for prediction of time series data based on a Support Vector Regression (SVR) algorithm with RBF loss function. We propose a strategy to build diverse sub-models of the ensemble based on the Feature Vector Selection (FVS) method of Baudat & Anouar (2003), which decreases the computational burden and keeps the generalization performance of the model. A simple but effective strategy is used to calculate the weights of each data point for different sub-models built with RBF-SVR. A real case study on a power production component is presented. Comparisons with results given by the best single SVR model and a fixed-weights ensemble prove the robustness and accuracy of the proposed ensemble approach.

Key words: Ensemble, Dynamic weighting, Feature Vector Selection, RBF kernel function, Nuclear power plant

1. INTRODUCTION

Combining various data-driven approaches into an ensemble is a relatively recent direction of research, aimed at improving the robustness and accuracy of the final prediction. The models which compose the ensemble are called sub-models. Various strategies have been proposed for building sub-models, including error-correcting output coding, Bagging, Adaboost, and Boosting (Kim, Pang, Je, Kim & Bang, 2003; Hu, Youn, Wang & Yoon, 2012). Similarly, several methods for aggregating the prediction results of the sub-models have been proposed, such as majority vote, weighted vote, Borda count, Bayes and probabilistic schemes, etc (Polikar, 2006).

Support Vector Machine (SVM) is a popular and promising data-driven method for prognostics. SVM-based ensemble models have been proposed for classification. Chen, Wang and Zuylen (2009) use ensemble of SVMs to detect traffic incidents. The sub-models use different kernel functions and parameters, and their outputs are combined to improve the classification performance. Acar and Rais-Rohami (2009) treat the general weighted-sum formulation of an ensemble as an optimization problem and, then, minimize an error metric to select the best weights for the sub-models of SVM. Kurram and Kwon (2013) try to achieve an optimal sparse combination of the sub-model results by jointly optimizing the separating hyperplane obtained by each SVM classifier and the corresponding weights of the sub-decisions. Valentini and Dietterich (2003) prove that an ensemble of SVMs employing bagging of low-bias algorithms improves the generalization power of the procedure with respect to single SVM. The ensemble of SVMs built with bagging and boosting can greatly outperform a single SVM in terms of classification accuracy (Kim *et al*., 2003).

In this paper, we focus on the combination of multiple SVR sub-models (Liu, Seraoui, Vitelli & Zio, 2012) with Radial Basis loss Function (RBF). The case study considered to present the application of the method concerns the monitoring of the leak flow in the first seal of the Reactor Coolant Pump (RCP) of a Nuclear Power Plant (NPP), using real data collected from sensors.

An ensemble of SVRs with RBF and dynamic weighting strategy is proposed in this paper. The elements of novelty of the method here proposed are various.

In the previously mentioned literature that works on ensembles of SVMs, the weights of the sub-models in the ensemble are calculated during training and kept fixed for testing. However, a sub-model may perform well only on a part of the dataset. Hence, the weights need to be updated considering the different datasets involved in the case study, and even different input

vectors. A dynamic weighting strategy, based on local fitness calculation (Baudat & Anouar, 2003) is proposed in this paper. A dynamic weighting method is also used in Muhlbaier, Topalis and Polikar (2009), Yang, Yuan and Liu (2009) and Razavi-Far, Baraldi and Zio (2012), for adding new classifiers to the ensemble model, but the weights are not adjusted to the different input vectors.

Moreover, in order to build an ensemble of SVRs on very large datasets, FVS is used to select a smaller subset of the training data points of each sub-model to decrease the computational burden.

In addition, a weighted-sum strategy is used to combine the outputs of the sub-models.

Finally, a strategy is proposed to form the training dataset of each sub-model based on the angle between different data points in the Reproducing Kernel Hilbert Space (RKHS).

All the novel strategies are tested in the case study concerning the monitoring of leak flow of the RCP in a NPP.

The rest of the paper is organized as follows. Section 2 gives details about the proposed ensemble approach. Section 3 illustrates the case study, the available data and how the proposed ensemble model is constructed. Section 4 presents the experimental results from the SVR ensemble models and describes the comparison with a single SVR model and a fixed weighted ensemble. Finally, conclusions with some considerations are drawn in Section 5.

2. DYNAMIC-WEIGHTED RBF-BASED ENSEMBLE

The underlying strategy motivating the use of ensemble-based methods in prediction problems is to benefit from the strength of different sub-models by combining their outputs to improve the global prediction performance, if compared to the results of a single sub-model.

In this section, we give details about the proposed Dynamic-Weighted RBF-based Ensemble (named DW-RBF-Ensemble, in short).

2.1 Standard Support Vector Regression with RBF and ε -sensitive loss function

Suppose a set of training data points $(\boldsymbol{x}_i, y_i)$, for $i = 1, 2, \dots, T$ is available. The construction of an SVR model amounts to finding the best estimate function $f(\boldsymbol{x}) = \boldsymbol{\omega x} + b$ of the real underlying function. To this aim, the primal quadratic optimization problem is

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

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

where ξ_i and ξ_i^* are slack variables. The dual formulation of Eq. (1) is

$$L = \frac{1}{2}\|\boldsymbol{\omega}\|^2 + C\sum_{i=1}^{T}(\xi_i + \xi_i^*) - \sum_{i=1}^{T}\alpha_i(\varepsilon + \xi_i - y_i + \boldsymbol{\omega x_i} + b) - \sum_{i=1}^{T}\alpha_i^*(\varepsilon + \xi_i^* + y_i - \boldsymbol{\omega x_i} - b) - \sum_{i=1}^{T}(\eta_i\xi_i + \eta_i^*\xi_i^*), \quad (2)$$

where $\alpha_i,\ \alpha_i^*,\ \eta_i,\ \eta_i^* \ge 0$ are the Lagrange multipliers. By calculating the partial derivative of L with respect to the primal variable $\boldsymbol{\omega}$, the best estimate function can be written as

$$f(\boldsymbol{x}) = \sum_{i=1}^{T}(\alpha_i - \alpha_i^*) * k(\boldsymbol{x}_i, \boldsymbol{x}) + b. \quad (3)$$

The values of α_i and α_i^* can be calculated by solving the Kuhn-Tucker conditions related to Eq. (2)

The kernel function $k(\boldsymbol{x}_i, \boldsymbol{x})$ in Eq. (3) enables the mapping of an input vector in a higher-dimensional RKHS. By calculating pairwise inner products between mapped samples, the kernel functions return the similarity between different samples. In fact, only kernels that fulfill Mercer's Theorem (i.e. the kernel matrix must be positive semi-definite) are valid ones and, thus, can be used in SVM (Minh, Niyogi and Yang, 2006). The most common kernel functions include the linear kernel function, the polynomial kernel function and the RBF. In this paper, the ensemble approach is proposed to be built based on SVR with RBF.

For RBF, $k(\boldsymbol{x}_i, \boldsymbol{x}) = e^{-\frac{\|x_i - x\|^2}{2\sigma^2}}$ and a good property for RBF is that for each data point $\boldsymbol{x}$, $k(\boldsymbol{x}, \boldsymbol{x}) = 1$, i.e., the data point in RKHS is a unit vector. The difference between different data points in RKHS is only the angle between them.

2.2 Feature Vector Selection

In Baudat and Anouar (2003), the authors propose a Feature Vector Selection (FVS) method to select a subset of the training data points (i.e. Feature Vectors (FVs)), which can represent the dimension of the whole dataset in RKHS. The other data points can all be expressed as a linear combination of the selected FVs.

Initialization:
Training dataset: $\boldsymbol{T}_r = \{(\boldsymbol{x}_i, y_i)\}$, for $i = 1, 2, \dots, T$
Feature space: **S** = [ ]
Threshold of LF: τ
FVS:
First FV in **S**:
For i = 1 to T, calculate
S = $\{\boldsymbol{x}_i\}$, compute the global fitness J_S for all training data points with respect to the present **S**.
End for.
Select the point which gives the maximum global fitness as the first FV and add it to **S** as the first FV.
$\boldsymbol{T}_r$ is reduced to the complement of **S** in $\boldsymbol{T}_r$, i.e. $\boldsymbol{T}_r = \boldsymbol{T}_r \backslash \mathbf{S}$.
Second and the other FVs:
Calculate the local fitness for all data points in $\boldsymbol{T}_r$ with respect to the present feature space **S**.
Select the data point k which gives the maximum of the local fitness:
If $1 - J_{S,k} > \tau$, this point is a new FV and is added to **S**; and $\boldsymbol{T}_r = \boldsymbol{T}_r \backslash \mathbf{E}$, with **E**=$\{ (\boldsymbol{x}_i, y_i) : 1 - J_{S,i} \leq \tau$ and $(\boldsymbol{x}_k, y_k)\}$;
If $1 - J_{S,k} \leq \tau$, end the process of FVS.

Fig. 1. Pseudo-code for 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 vector $\boldsymbol{x}_i$ into RKHS with the mapping $\boldsymbol{\varphi}_i$, for $i = 1, 2, \dots, T$. The kernel $k_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the inner product between $\boldsymbol{\varphi}_i$ and $\boldsymbol{\varphi}_j$. Suppose that the FVs selected from the training dataset are $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}$ and the corresponding mapping is S = $\{\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2, \dots, \boldsymbol{\varphi}_N\}$: the process for selecting the new next FV is to calculate $\{a_{new,1}, a_{new,2}, \dots, a_{new,T}\}$ which gives the minimum of Eq. (4), with $\boldsymbol{\varphi}_{new}$ being the mapping of the new input vector $\boldsymbol{x}_{new}$:

$$\delta_{new} = \frac{\left\| \boldsymbol{\varphi}_{new} - \sum_{i=1}^{L} a_{new,i} \boldsymbol{\varphi}_i \right\|^2}{\|\boldsymbol{\varphi}_{new}\|^2}. \quad (4)$$

The minimum of δ_{new} can be expressed with an inner product, as shown in Eq. (5):

$$\min \delta_{new} = 1 - \frac{K_{S,new}^t K_{S,S}^{-1} K_{S,new}}{k_{new,new}}, \quad (5)$$

where $K_{S,S} = (k_{i,j}), i, j = 1,2, \dots, N$ is the kernel matrix of S and $K_{S,new} = (k_{i,N}), i =$

$1,2,\dots,N$ is the vector of the inner product between $\boldsymbol{\varphi}_{new}$. The expression $J_{S,new} = \frac{K_{S,new}^{t} K_{S,S}^{-1} K_{S,new}}{k_{new,new}}$ is the local fitness of $\boldsymbol{x}_{new}$ with respect to the present feature space S. If $1 - J_{S,new}$ is zero, the new data point is not a new FV; otherwise, it is a new FV and is added to S. With the global fitness defined as in Eq. (6), the FVS procedure proceeds to select a subset of training data points with minimal size, which gives zero global fitness. The details for FVS is shown in Figure 1.

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

2.3 Ensemble-Based Approach

An ensemble-based approach is obtained by training diverse sub-models and, then, combining their results following given strategies. It can be proven that this can lead to superior performance with respect to a single model approach (Bauer & Kohavi, 1999). A simple paradigm of a typical ensemble-based approach with *N* sub-models is shown in Figure 2. Ensemble models are built on three key components: a strategy to build diverse models; a strategy to construct accurate sub-models; a strategy to combine the outputs of the sub-models in a way such that the correct predictions are weighted more than the incorrect ones.

In the DW-RBF-Ensemble that we are proposing, the sub-models are built using a modified SVR model with RBF.

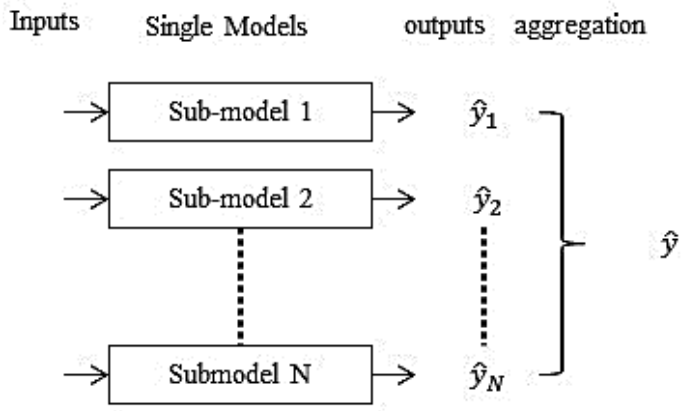

Fig. 2. Paradigm of a typical ensemble method.

A dynamic weighted-sum strategy is proposed to combine the outputs of the sub-models. As mentioned in the Introduction, different methods can be applied to calculate the weights for the sub-models. In the methods that can be found in the literature, the weights are normally fixed after the ensemble model is built. They are only updated when new sub-models are added to the ensemble or when some sub-models are changed. In some real applications with fast

changing environmental and operational conditions, the performance of the ensemble model may degrade rapidly. This degradation is not always caused by the low robustness or capability to adapt of the ensemble model, but can be due to the fact that the best sub-models are not given proper weights.

In this paper, a dynamic weighting strategy is thus proposed. The weights are no longer constant during the prediction, but dependent on the input vector. They are recalculated each time a new input vector arrives. Inspired by the work of Baudat and Anouar (2003) and considering the characteristics of SVR, a local fitness calculation is implemented in this paper to calculate the weights of the different sub-models for each input vector.

2.3.1 Sub-datasets determination

Clustering methods are widely used in ensemble approaches for determining the sub-datasets for different sub-models.

In this paper, SVR models are trained with RBF. The difference between different data points in RKHS is only the angle between them, as the norm of all data points in RKHS is one. Thus, we can use the angular-clustering algorithm to divide the whole training dataset into several sub-datasets. The pseudo-code is shown in Figure 3. As kernel function, RBF is the inner product of two vectors in RKHS and the angle between them can be expressed as Eq. (7) in the pseudo-code of Figure 3.

For training dataset $\boldsymbol{T} = \{(\boldsymbol{x}_i, y_i)\}, i = 1,2, \dots, T$, choose the number of clusters c. Initialize a random cluster centers vector $(\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_c)$ chosen from the dataset $\boldsymbol{T}$.

Repeat: for $l = 1,2, \dots$

Compute the angle

$$D_{ik} = \arccos(k(\boldsymbol{x}_k, \boldsymbol{v}_i^{(l)})), 1 \le i \le c,$$

$1 \le k \le T$. $k(\cdot,\cdot)$ is the RBF. (7)

Select the points of minimal distances for a cluster, and they belong to that cluster. Suppose $\{(\boldsymbol{x}_i, y_i)\}, i = 1,2, \dots, N_i$ are data points in cluster i,

Calculate the cluster centers

$$\boldsymbol{v}_i^{(l+1)*} = \frac{\sum_j^{N_i} \boldsymbol{x}_j}{N_i}.$$

Choose the nearest data point to the calculated cluster center in the corresponding cluster to be the new cluster center.

$$D_{ik}^* = \arccos\left(k\left(\boldsymbol{x}_k, \boldsymbol{v}_i^{(l+1)*}\right)\right), 1 \le k \le$$

N_i and $\boldsymbol{v}_i^{(l+1)} = argmin_{\ i}(D_{ik}^*)$.

Until $|D^{(l+1)} - D^{(l)}| \le \tau, \tau$ is a small positive value and $D^{(l)} = \sum_{i=1}^{c} \sum_{k=1}^{T} D_{ik}$.

Fig. 3 Pseudo-code of angle-clustering algorithm.

2.3.2 Train a RBF-SVR sub-model

With the angle-clustering method, the training dataset is divided into several clusters. But in the DW-RBF-Ensemble method, the data points in each cluster are not used directly to train a RBF-SVR. FVS is firstly used to select the FVs in each cluster and, then, the SVR model is trained on these selected FVs, in order to decrease the computational burden. The procedures for training a SVR model with FVs are not the same as shown in Sub-Section 2.1, as the estimate function in Eq. (2) is no longer a kernel expansion on all the training data points in one cluster, but only on the selected FVs.

Suppose that for the j-th cluster, the training data points are $(\boldsymbol{x}_i, y_i)$, for $i = 1, 2, \dots, T_j$ and the FVs selected by FVS are $(\boldsymbol{x}_i, y_i)$, for $i = 1, 2, \dots, N_j$; the estimate function of SVR for the i-th cluster is given in Eq. (8):

$$f(\boldsymbol{x}) = \sum_{i=1}^{N_j} (\alpha_i - \alpha_i^*) * k(\boldsymbol{x}_i, \boldsymbol{x}) + b. \quad (8)$$

In order to avoid the overfitting problem, the optimization still aims at finding the minimum of

the objective function in Eq. (1) on all the training data points in the cluster. Thus, by replacing $\boldsymbol{\omega x_i} + b$ in Eq. (2) with $\sum_{k=1}^{N_j}(\alpha_k - \alpha_k^*) * k(\boldsymbol{x}_k, \boldsymbol{x_i}) + b$, we can have the new, dual formulation of SVR. Classical methods can be used to estimate the unknowns in Eq. (8).

Such a process can efficiently decrease the risk of overfitting and guarantee the generalization performance of the sub-models.

2.3.3 Weights Calculation

In Section 2.2, FVS defines global and local criteria to characterize the feature space. The proposed local fitness can describe the linearity between the mapping of a new input vector and the mapping of all the Feature Vectors (FVs) of the model: if a linear combination of the mapping of the FVs can better approach the mapping of the new input vector, i.e. $1 - J_{S,new} \approx 0$ the model gives better approximation of the output of the new data point; otherwise, i.e. $1 - J_{S,new} \approx 1$, the model performs worse for this data point. Thus local fitness can be implemented to derive the weight of each sub-model for each input vector.

With Eq. (5), for a new coming data point at time *t*, we can calculate the local fitness $J_i(t)$ with respect to the FVs of the *i*-th sub-model. And the weight of the *i*-th sub-model for this data point is calculated as

$$\omega_i(t) = \frac{1/(1-J_i(t)+\tau)}{\sum_{j=1}^{N} 1/(1-J_j(t)+\tau)}, \quad (9)$$

where τ is a very small value so that Eq. (9) works in the case $J_i(t) = 1$.

2.3.4 Combining Sub-Models Outputs

Figure 4 shows the paradigm of DW-RBF-Ensemble, where N is the number of sub-models, $\boldsymbol{x}(t)$ is a new input vector arriving at time t, $w_j(t)$ is the weight assigned to the *j*-th sub-model for the new input vector, $\hat{y}_j(t)$ is the predicted value for the *j*-th sub-model given by RBF-SVR and $\hat{y}(t)$ is the final output of the ensemble model.

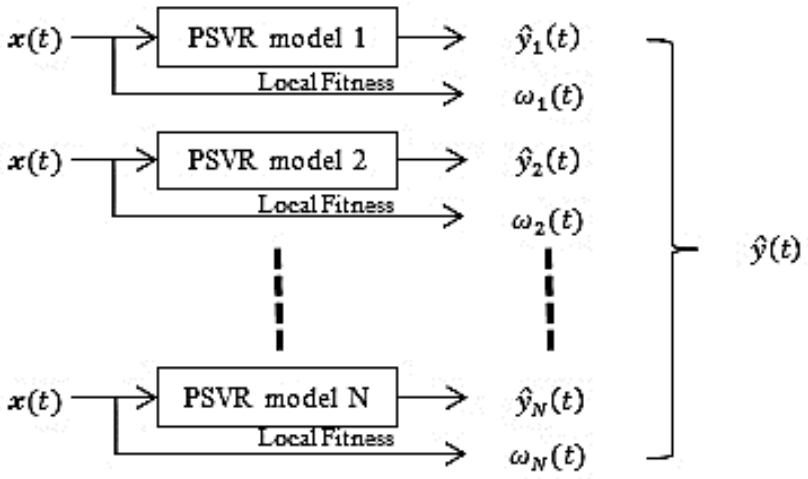

Fig. 4. Paradigm of the proposed DW-PSVR-Ensemble.

We can derive the fact that $\hat{y}(t) = \sum_{j=1}^{N} \omega_j(t)\hat{y}_j(t)$, if we assume sub-models results to be uncorrelated.

Note that all the sub-models weights and outputs are a function of t, which means that they are all dependent on the input vector of the ensemble model.

3. CASE STUDY DESCRIPTION

The real case study considered in this paper concerns the 1-day ahead prediction of leak flow from the first seal of the RCP of a NPP.

In this section we describe the time series data and briefly recall the data pre-processing steps. We also detail the strategies to build the diverse sub-models of the ensemble.

3.1 Data Description and Pre-Processing

The data provided correspond to 9 scenarios of leak flow from different NPPs. Each scenario contains a time series data of the leak flow. They are named Scenario 1, Scenario 2, …, Scenario 9 in the following sections of the paper. These data are monitored every four hours. As these data are time-dependent and recorded within different time windows, only scenarios coming from the same NPP have the same size. In some of the scenarios, there are missing data points and outliers.

Since the dataset we are going to analyze contains both missing data and outliers, we have to deal with both these issues. First of all, we must remove anomalous data, since their extreme values would affect the results of the analysis. Outliers can be detected with reference to some constraints, e.g. the limits $\bar{x} \pm 3 * \sigma_x$ where $\bar{x}$ is the mean of the data points values and σ_x

is the standard deviation. These limits allow detect the outliers, selected as those data points whose values are larger than $\bar{x} + 3 * \sigma_x$ or smaller than $\bar{x} - 3 * \sigma_x$, and subsequently removed. Note that we use such constraints, rather than the usual ones based on the median and the InterQuartile Range (IQR), to be more conservative in the outlier selection, due to the dependence among data (Brodsky, Lemmens, Brock-Utne, Vierra & Saidman, 2002).

Secondly, we want to reconstruct missing data. A possible way to deal with the reconstruction of missing data is local polynomial regression fitting (Masry, 1996). This local least squares regression technique estimates effectively the values of missing data points. Moreover, it can also be used to perform the smoothing of the available observations, in order to reduce noise. We will, thus, use this technique both to reconstruct data where missing, and to obtain a smoother and less noisy time series in all remaining time instances. All the time series data of all scenarios are, then, normalized from 0 to 1. All details on this pre-processing task can be found in Liu *et al.* (2012).

3.2 Strategies to Build Sub-Models

We have a time series dataset and we need to decide the best number of historical values to be used as inputs.

Suppose $a(t)$ represents an instance of the time series data of one scenario. For 1-day ahead prediction, the output $y(t)$ is $a(t+6)$, because the signals are monitored every four hours. In order to decide the best H for selecting the input vector $\boldsymbol{x}(t) = (a(t-H+1), \dots, a(t))$ most related to the output, a partial autocorrelation analysis is carried out, i.e. the correlation between the output values at current time and different temporal lags is computed. Figure 3 shows the results of this analysis on all the scenarios, where the *x* and *y* axis represent the temporal lag (a multiple of four hours) and the corresponding empirical partial autocorrelation, respectively. The bounds of a 95% confidence interval are also shown with dashed lines in the Figure. The correlation decreases with the lag (although not linearly) and after a lag of 17 time steps it is no longer comparable with the values observed for lags smaller than 17, i.e. the best choice is $H = 17$.

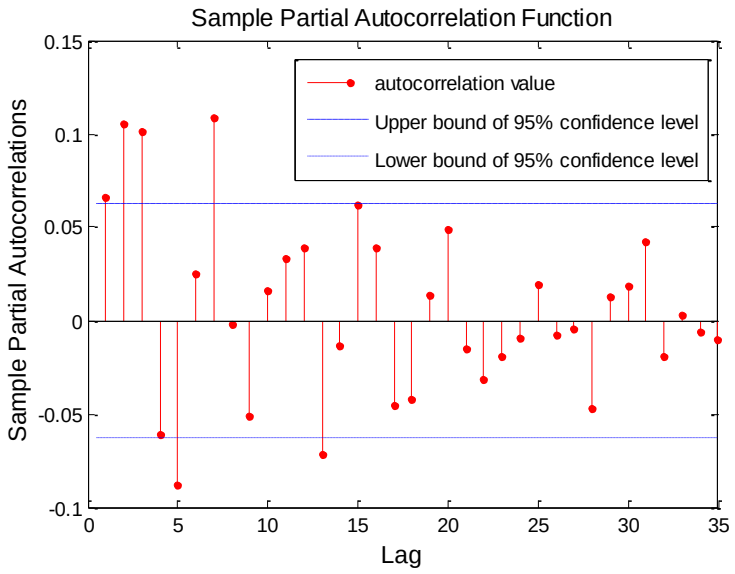

Figure 3. Partial autocorrelation function with respect to time lags (multiples of four hours). Dotted lines are the bounds of the 95% confidence interval.

Then, the training dataset is divided into several sub-datasets for different sub-models using the angle-clustering algorithm described in sub-section 2.3.1.

3.3 Comparison of DW-RBF-Ensemble with Single SVR and Fixed Weights Ensemble

The ensemble model is expected to give better results than a single SVR model. To verify this claim, a comparison between a single SVR model and the proposed DW-RBF-Ensemble is carried out on the considered case study. A fixed weights ensemble (Kurram and Kwon, 2013) is also taken as a benchmark method to prove the benefit of using a dynamic weighting strategy.

Each time one out of 9 scenarios is chosen as the test dataset (named Observed Scenario) and the other 8 scenarios (named Reference Scenarios) from the training dataset which is used to construct the DW-RBF-Ensemble and the Fixed Weights Ensemble (FW-Ensemble). A SVR model is also trained on the training dataset for comparison (it is named Single SVR to be distinguished from the two ensemble models).

The steps for the comparison are the following:

1. Train a Single SVR model with all the training dataset.
2. The training dataset is divided into 6 clusters by the angle-clustering algorithm.
3. Train DW-RBF-Ensemble: FVS select the FVs in each cluster and a sub-model is trained on the selected FVs. Weights of different sub-models for each data point are calculated with Eq. (9).

4. Train a FW-Ensemble: train a sub-model with all the data points in each cluster. The weight for each sub-model is decided by minimizing the MAE on the training dataset.
5. Calculation of Mean Absolute Error (MAE), Mean Relative Error (MRE) of the outputs of DW-RBF-Ensemble, FW-Ensemble and Single PSVR.
6. Compare prediction accuracy, computational burden and model robustness.

The results and comparisons among these models are presented in the next section.

4. RESULTS

In this section, the results from DW-RBF-Ensemble, FW-Ensemble and Single SVR are compared with respect to different aspects.

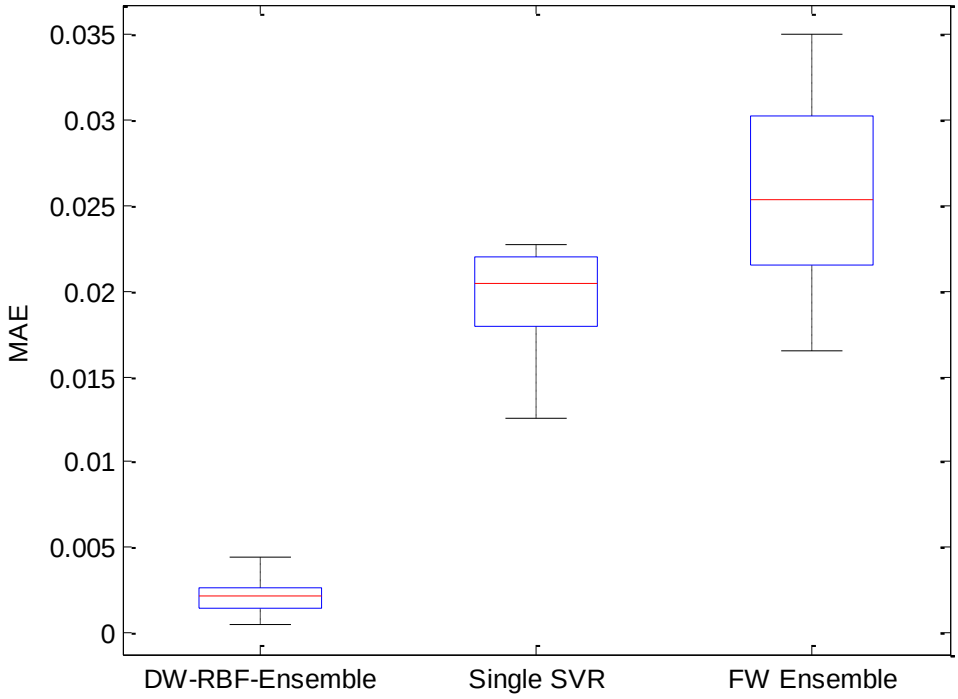

Figure 4. MAE of prediction results of ensembles and Single SVR, for all 9 scenarios.

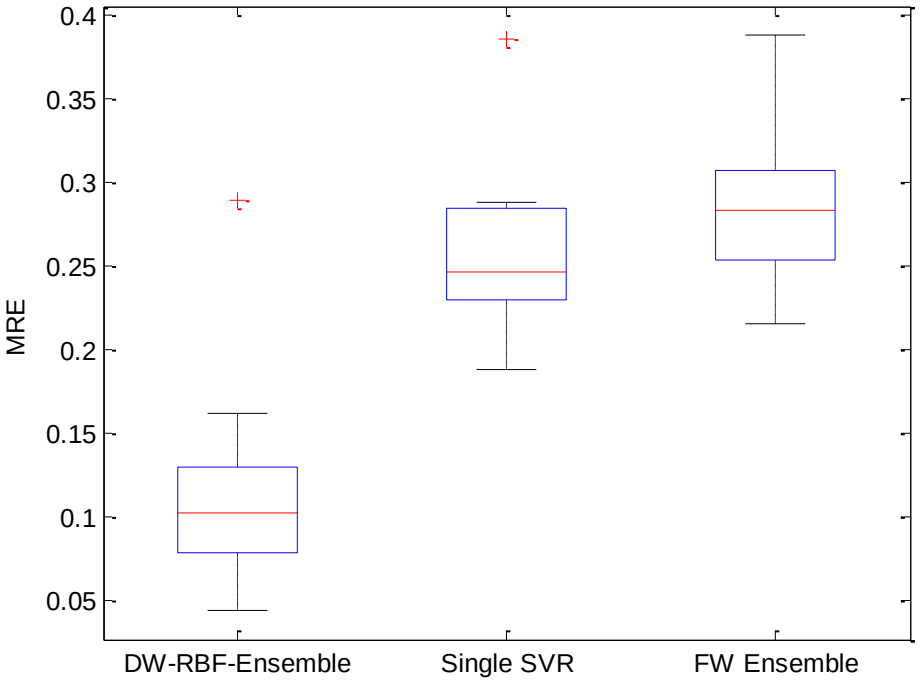

Figure 5. MRE of prediction results of ensembles and Single SVR, for all 9 scenarios.

4.1 Prediction Accuracy

Figures 4 and 5 report the prediction results of MAE and MRE obtained, respectively, by DW-RBF-Ensemble, FW-Ensemble and Single SVR. It is clear that DW-RBF-Ensemble gives best results in this case study, i.e. on average, the MAE and MAE values are smaller than for Single SVR and FW-Ensemble.

The bad results of the Single SVR are caused by the fact that the predictions are highly dependent on the training dataset. Moreover, the hyperparameters optimization is also critical to the performance of SVR. Well-chosen hyperparameters values can improve the performance of the SVR. However, the optimization method may converge to a local extreme, which results into a good performance at the beginning but bad at the end of the scenario. The ensemble approach can avoid such problem by combining the results from different sub-models.

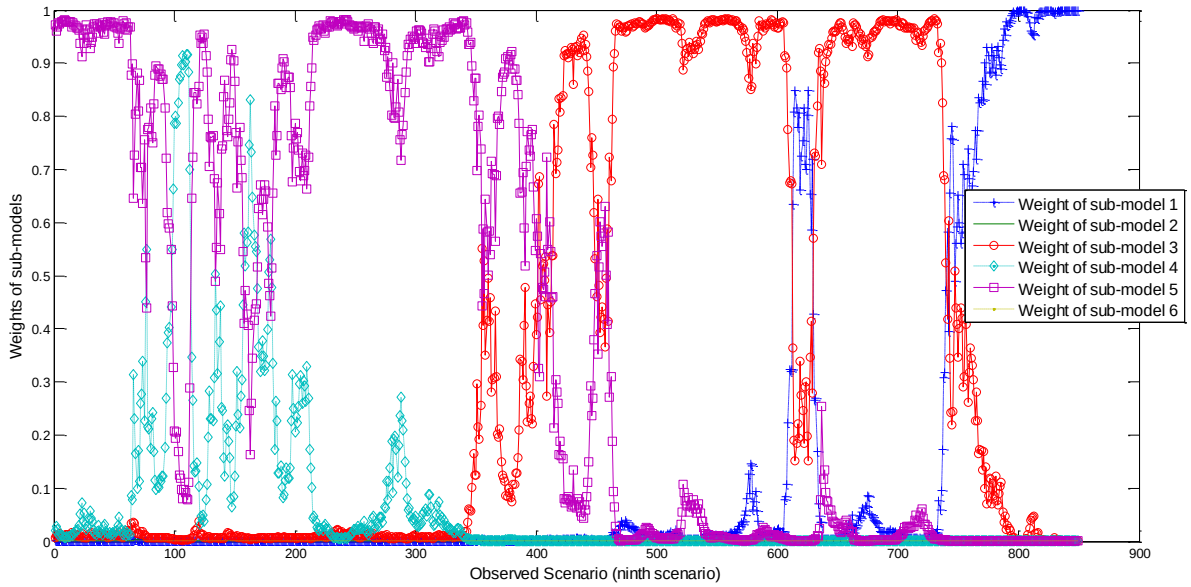

Figure 6. Weights of different sub-models of DW-RBF-Ensemble for test data points of the ninth scenario.

These unstable results from the Single SVR prove the necessity of the ensemble approach for avoiding the limits of Single SVR in attaining the desired accuracy and robustness of the model.

In this case study, FW-Ensemble gives the worst results as the weights are fixed after training. For some data points, the best model is not given the most important weight. Figure 6 above shows the weights for different sub-models of DW-RBF-Ensemble in the case of selecting the ninth scenario as the Observed Scenario. It is clear that the weights of the sub-models change frequently to adapt to the ongoing data points.

The prediction results from DW-RBF-Ensemble confirm the practicability and efficiency of the proposed approach.

4.2 Robustness

From Figures 4 and 5, it is seen that the DW-RBF-Ensemble gives more stable prediction results compared to the Single SVR model and FW Ensemble. The Single SVR model cannot properly handle the noise in the data and it is difficult to find the global optimal values of the hyperparmeters. The weighted-sum ensemble models can decrease the influence of the noise by combining the prediction outputs of the sub-models. But the fixed weighting strategy cannot adapt to the changing environment and the weights of the sub-models are not changed adaptively. This is one reason for which DW-RBF-Ensemble model can give stable results, i.e. the DW-RBF-Ensemble model is more robust compared to the Single SVR and FW-Ensemble.

4.3 Computational complexity

Suppose the size of the training dataset is T; then, the computational complexities of the Single SVR for training and testing are T^3 and T, respectively. For very large datasets, the computational burden of the Single SVR model is very high and sometimes unacceptable. By dividing the training dataset into different sub-datasets, the total computational burden is decreased as $T^3 > T_1^3 + \cdots + T_N^3$, with $T_1 + \cdots + T_N = T$. With FVS, the size of the training dataset is further decreased for training and testing. Thus, the computational complexity of the DW-RBF-Ensemble approach is much smaller than the Single SVR trained on all the training dataset and the FW-Ensemble.

5. CONCLUSIONS

In this paper, we have proposed an innovative dynamic-weighted RBF-based ensemble approach for short-term prediction (1-day ahead prediction) with time series data. An angular-clustering algorithm is used to divide the training dataset into sub-datasets and FVS is used to decrease the size of the training data points by selecting only the representative data points in RKHS. Local fitness calculation is integrated to calculate the specific weights of the sub-models of the ensemble for each new input vector, without bringing too much computational burden.

The proposed ensemble approach has been shown to perform well in a real case study of signals recorded on a NPP component. Compared to the single SVR model and FW Ensemble, the proposed ensemble model outperforms them on prediction accuracy, computational burden, robustness and adaptability.

Further research needs to be carried out for optimizing the numbers of sub-models and for obtaining a more careful tuning of the hyperparameters.

REFERENCES

Acar, E., & Rais-Rohani, M. (2009). Ensemble of metamodels with optimized weight factors. *Structural and Multidisciplinary Optimization*, *37*(3), 279-294.

Bauer, E., & Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning*, *36*(1-2), 105-139.

Baudat, G., & Anouar, F. (2003). Feature vector selection and projection using kernels. *Neurocomputing*, *55*(1-2), 21-38.

Brodsky, J. B., Lemmens, H. J., Brock-Utne, J. G., Vierra, M., & Saidman, L. J. (2002). Morbid obesity and tracheal intubation. *Anesthesia & Analgesia*,*94*(3), 732-736.

Chen, S., Wang, W., & Van Zuylen, H. (2009). Construct support vector machine ensemble to detect traffic incident. *Expert systems with applications*, *36*(8), 10976-10986.

Gao, J. B., Gunn, S. R., Harris, C. J., & Brown, M. (2002). A probabilistic framework for SVM regression and error bar estimation. *Machine Learning*,*46*(1-3), 71-89.

Gurram, P., & Kwon, H. (2013). Sparse kernel-based ensemble learning with fully optimized kernel parameters for hyperspectral classification problems.*Geoscience and Remote Sensing, IEEE Transactions on*, *51*(2), 787-802.

Hu, C., Youn, B. D., Wang, P., & Taek Yoon, J. (2012). Ensemble of data-driven prognostic algorithms for robust prediction of remaining useful life. *Reliability Engineering & System Safety*, *103*, 120-135.

Kim, H. C., Pang, S., Je, H. M., Kim, D., & Yang Bang, S. (2003). Constructing support vector machine ensemble. *Pattern recognition*, *36*(12), 2757-2767.

Liu, J., Seraoui, R., Vitelli, V., & Zio, E. (2013). Nuclear power plant components condition monitoring by probabilistic support vector machine. *Annals of Nuclear Energy*, *56*, 23-33.

Masry, E. (1996). Multivariate local polynomial regression for time series: uniform strong consistency and rates. Journal of Time Series Analysis, 17(6), 571-599.

Minh, H. Q., Niyogi, P., & Yao, Y. (2006). Mercer's theorem, feature maps, and smoothing. In *Learning theory* (pp. 154-168). Springer Berlin Heidelberg.

Muhlbaier, M. D., Topalis, A., & Polikar, R. (2009). Learn. NC: Combining Ensemble of Classifiers With Dynamically Weighted Consult-and-Vote for Efficient Incremental Learning of New Classes. *Neural Networks, IEEE Transactions on*, *20*(1), 152-168.

Polikar, R. (2006). Ensemble based systems in decision making. *Circuits and Systems Magazine, IEEE*, *6*(3), 21-45.

Razavi-Far, R., Baraldi, P., & Zio, E. (2012). Dynamic Weighting Ensembles for Incremental Learning and Diagnosing New Concept Class Faults in Nuclear Power Systems. *Nuclear Science, IEEE Transactions on*, *59*(5), 2520-2530.

Smola, A. J., & Schölkopf, B. (2004). A tutorial on support vector regression. Statistics and computing, 14(3), 199-222.

Sollich, P. (1999). Probabilistic interpretations and Bayesian methods for support vector machines. In *Artificial Neural Networks, 1999. ICANN 99. Ninth International Conference on (Conf. Publ. No. 470)* (Vol. 1, pp. 91-96). IET.

Valentini, G., & Dietterich, T. G. (2003, August). Low bias bagged support vector machines. In *ICML* (pp. 752-759).

Yang, X., Yuan, B., & Liu, W. (2009, November). Dynamic Weighting Ensembles for incremental learning. In *Proc. of IEEE conference in pattern recognition* (pp. 1-5).

PAPER IV: JIE LIU & ENRICO ZIO "AN ADAPTIVE ONLINE LEARNING APPROACH FOR SUPPORT VECTOR REGRESSION," *IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS*, 2014. (UNDER REVIEW)

AN ADAPTIVE ONLINE LEARNING APPROACH FOR SUPPORT VECTOR REGRESSION

Jie Liu, and Enrico Zio, *Senior Member, IEEE*

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 nonstationary 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 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, MRE and NMSE 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

Many 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 model 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 nonstationary 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 recent special issue of the journal IEEE Transactions on Neural Networks and Learning Systems provides an interesting up-to-date snapshot of the ongoing research in this area (see for examples from that special issue, the papers in [20], [7], [20], [13], [6], [22]).

Support Vector Regression (SVR) is one of the most popular data-driven approaches. However, it also faces the problem of changing environments, due to computational complexity with large datasets, and adaptation to pattern drifts. Some approaches have been proposed in the literature for SVR to adaptively learn new data points. In these approaches, the online learning of a trained SVR model is mostly based on prediction accuracy and/or characteristics of the inputs of the data points. The rationale is to add data points as basis for the SVR model when they are not predicted well and/or contain new information on the input space. Reference [18] proposes a novel 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 a novel 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 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] propose 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, as adding each new point available is time-consuming. Karasuyama and Takeuchi [11] propose a multiple incremental algorithm of SVM, based on the previous results. These 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 the methods consider 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, an adaptive online learning approach is proposed 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 combines a simplified version of the Feature Vector Selection (FVS) method introduced in [1] with Incremental & Decremental Learning presented in [4], considering the characteristics of the inputs and the bias of the prediction for the new data points. 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) 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 carefully 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 proposed Online-SVR-FID is similar to the method proposed in [23]. But the method proposed in [23] adds both the new patterns and changed patterns in the model, while the addition of changed patterns can destroy the nonlinear independence of the FVs in the model and, thus, ruin the FVS calculation in the following online learning procedure. In Online-SVR-FID, the changed patterns are used to replace one existing FV to keep the nonlinear independence, which is critical during online learning. A real case study is worked out concerning the leak flow from a seal of a pump in a Nuclear Power Plant (NPP). Comparisons 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 real case study with the experimental results and 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 $k(\boldsymbol{x}, \boldsymbol{x}_i) = e^{-\|x - x_i\|^2/2\sigma^2}$ 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 $i = 1, 2, \ldots, T$, 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 $i = 1, 2, \ldots, T$; $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \varphi(\boldsymbol{x}_i), \varphi(\boldsymbol{x}_j) \rangle$ 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 $\mathbf{S} = \{\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2, \ldots, \boldsymbol{\varphi}_L\}$ and the corresponding original data points are $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_L\}$. The verification of the new FV amounts to finding the vector $\boldsymbol{a}_N = \{a_{N,1}, a_{N,2}, \ldots, a_{N,L}\}$ 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 $K_{S,N} = (k_{i,N}), i = 1,2, \ldots, L$ 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

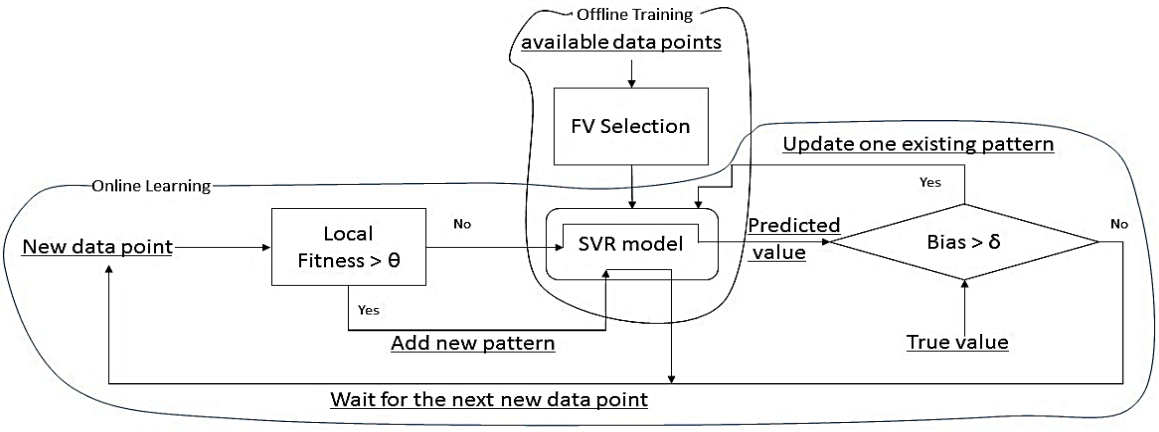

Fig. 2. Paradigm of Online-SVR-FID.

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. Fig. 1 is an example of a bi-

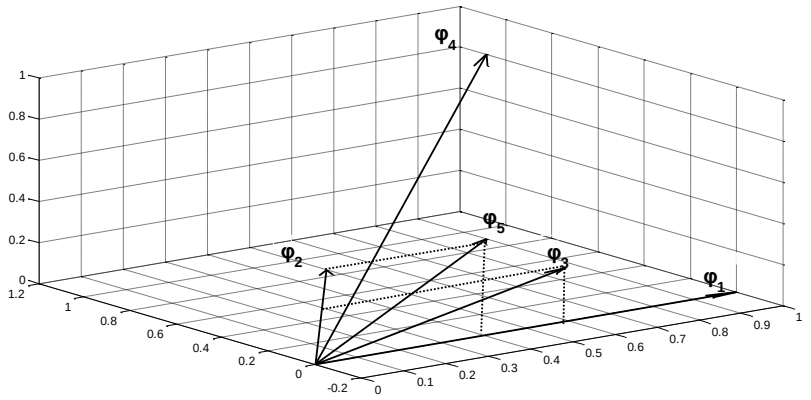

Fig. 1. Geometric explanation of FVS and local fitness in RKHS.

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.

2.3 Incremental & 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

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 Fig. 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 $J_{\boldsymbol{S}}$ 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 $\mathbf{E} = \{(\boldsymbol{x}_k, y_k)\ and\ (\boldsymbol{x}_i, y_i)\colon 1-J_{\boldsymbol{S},i} \leq \rho\}$; 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

Initialization:
Training dataset: $T_r = \{(x_i, y_i)\}$, for $i = 1, 2, ..., T$
Testing dataset: $T_e = \{(x_i, y_i)\}$, for $i = T+1, T+2, ..., T+H$
Feature space: S = [ ]
Threshold of local fitness: ρ
Threshold of bias: δ
Offline Training:
First FV in S:
For $i = 1$ to T calculate
$\mathbf{S} = \{x_i\}$, compute global fitness J_S.
End for.
Select the point which gives the maximum of the global fitness as the first FV and add it to S as the first FV.
T_r is reduced as the complement of S in T_r, i.e. $T_r = T_r \backslash S$.
Second and the other FVs:
Calculate local fitness for data points in T_r with the present feature space S;
Select the data point k which gives the minimum of local fitness;
If $1 - J_{S,k} > \rho$, this point is a new FV and added to S; $\mathbf{E} = \{(x_k, y_k)\ and\ (x_i, y_i)\colon\ 1 - J_{S,i} \leq \rho\}$ and T_r is reduced as the complement of E in T_r, i.e. $T_r = T_r \backslash \mathbf{E}$;
If $1 - J_{S,k} \leq \rho$, end the process of FVs selection;
Train the SVR model on the FVs in S.
Online Learning:
When a new data point (x_N, y_N) is available DO
Calculate the local fitness $J_{S,N}$ of this new data point;
If $1 - J_{S,N} > \rho$
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. Go back to the beginning of Online learning and wait for the next new data point.
If $1 - J_{S,N} \leq \rho$, verify the bias between the target of this new data point and the predicted value
If the bias is smaller than δ
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. (4). Unlearn this FV found with decremental learning 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.

Fig. 3. Pseudo-code of offline training & online learning using Online-SVR-FID.

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. $1 - J_{S,N} \leq \rho$, 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

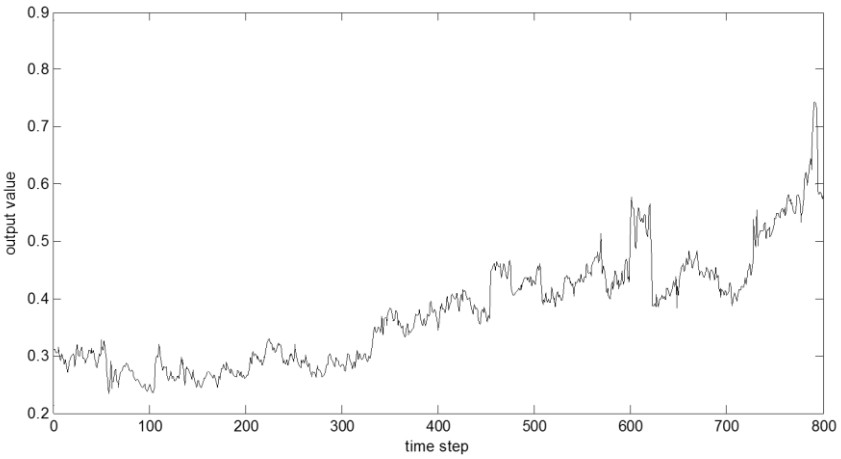

Fig. 4. Outputs of training (1-300) and testing (301-700) datasets.

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. REAL CASE STUDY

In this section, 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 safe operation of a NPP. Its function is to provide cooling water into the reactor, to extract the heat produced by nuclear

fission. 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].

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:

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

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 Mean Square Error (MSE) on the whole training dataset instead of only on the selected FVs.

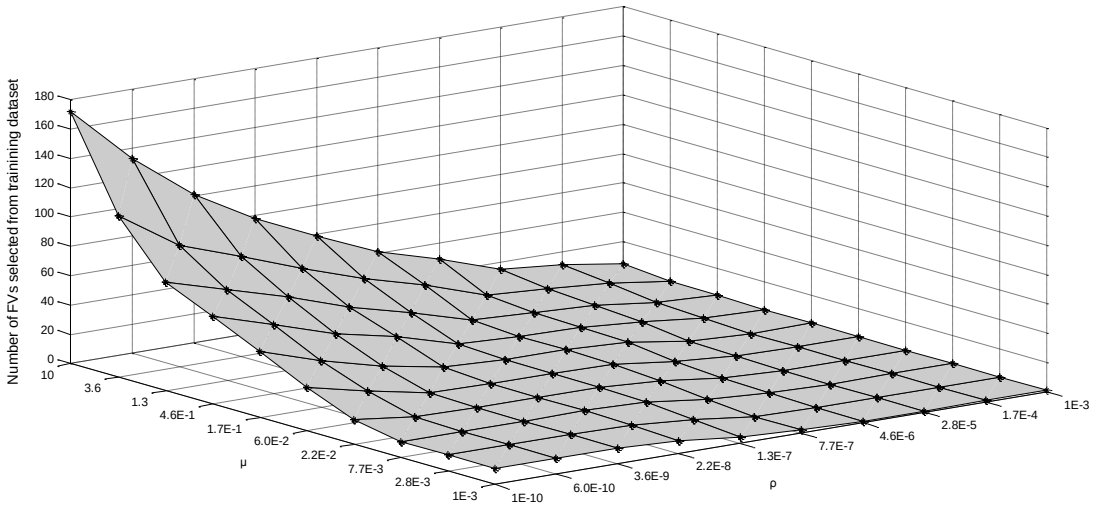

Fig. 5. Number of FVs selected from training dataset using different μ in (7) and ρ.

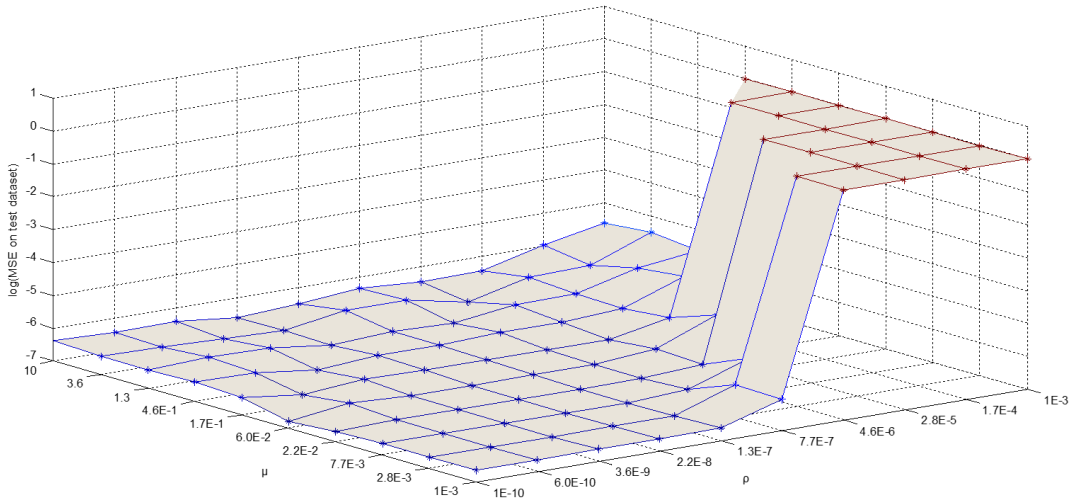

Fig. 6. MSE on the testing dataset using different μ in (7) and ρ.

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. 4 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

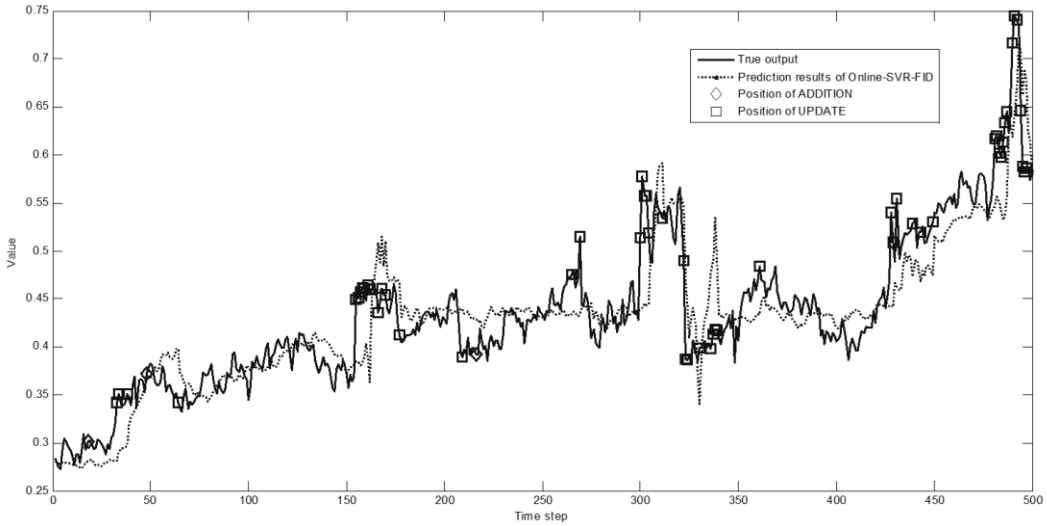

Fig. 7. Prediction results of the test dataset using Online-SVR-FID.

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. 5. Fig. 6 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. 6, 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.

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. 7 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.

4. COMPARISON WITH OTHER ONLINE LEARNING APPROACHES

In this section, Online-SVR-FID is compared with four other online learning methods: original Incremental Learning in [4], Naïve Online R_{reg} Minimization Algorithm (NORMA) in [12], SOGP in [3] and Kernel-based Recursive Least Square Tracker (KRLS-T) in [14].

4.1 Brief Introduction to the Online Learning Methods Considered

Incremental Learning is specifically developed for online learning of SVR [4]. It is designed so as to satisfy always the Kuhn-Tucker conditions, which are sufficient and necessary conditions for the solution of the quadratic programming problem in SVR. Such specific method is also integrated in the proposed Online-SVR-FID. In the experiment, Incremental Learning adds each new data point in the model instead of the selected ones.

NORMA considers classical stochastic gradient descent within the RKHS. NORMA performs gradient descent with respect to the instantaneous regularized risk on a single point, defined as the sum of the empirical risk and the complexity of the underlying function [12]. A learning rate η, which is usually kept constant during the learning iterations, is introduced to guarantee a bound on the number of operations required per iteration and the size of the model. NORMA can be used for regression, classification and novelty detection [12]. In this paper, it is applied to the nonlinear regression problem of interest for the case study of Section 3. When a new data $(\boldsymbol{x}_N, y_N)$ is available, NORMA adds this point to the model and updates its multipliers following (8), supposing β_N is the multiplier for the new data point in the support vector expansion of (2):

$$(\beta_i, \beta_N, \varepsilon) = \begin{cases} \big((1-C\eta)\beta_i, \eta|f(\boldsymbol{x}_N) - y_N|, \varepsilon + (1-\upsilon)\eta\big), & if\ |f(\boldsymbol{x}_N) - y_N| > \varepsilon \\ \big((1-C\eta)\beta_i, 0, \varepsilon - \upsilon\eta\big), & otherwise \end{cases}. \quad (8)$$

SOGP in [3] trains a model with a Bayesian online algorithm and, then, uses a FVs selection procedure to reduce the dimension of the Gaussian process by considering the geometrical relations in RKHS as FVS. SOGP tries to randomly delete the training data points that can be represented as a linear combination of the other training data points in RKHS. As it considers only the geometric relation between different input vectors in RKHS during the dimension

reduction, SOGP might ignore some data points with informative outputs in the case that different outputs exist for the same input vectors.

KRLS-T in [14] is a Bayesian perspective of the standard KRLS equations, which can perform tracking in nonstationary scenarios by forgetting in a consistent way under a fixed budget.

Details for these online learning approaches can be found in the related literature.

4.2 Comparison Results

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-T, 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, Mean Relative Error (MRE), Normalized Mean Squared Error (NMSE)) 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. I.

4.2.1 Computational Complexity

As a way to evaluate the computational complexity, we compare the times of Online Learning.

TABLE I

Comparisons of online prediction results with Online-SVR-FID, Incremental Learning, NORMA and SOGP

	Online-SVR-FID	Incremental Learning	NORMA	SOGP	KRLS
MSE	0.0011	0.0013	1.7091	0.0019	0.0044
MRE	0.0561	0.0548	2.9965	0.0763	0.0779
NMSE	0.0056	0.0069	8.854	0.0098	0.0228
Online Learning time (s)	9.2067	1354.6425	3191.8332	332.7395	9.5970
Model size before Online Learning	11	300	300	25	200
Model size after Online Learning	14	800	269	60	200

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. I) 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. This is because the Incremental & Decremental Learning in Online-SVR-FID is an iterative process while the adaptation of a KRLS-T is directly 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.

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.

4.2.2 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, 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.

5. CONCLUSIONS

In this paper, we have proposed an online learning approach for SVR, named Online-SVR-FID, to efficiently address by online learning the pattern drifts problem.

A real case study has been considered, concerning the prediction of the dynamic evolution of the leak flow from the first seal of a pump in a nuclear power plant for prognostic purposes. The approach is shown to be capable of significantly 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 with other online learning approaches i.e. NORMA, SOGA, KRLS and Incremental Learning, considering MSE, MRE and NMSE on the test dataset, Online-SVR-FID has been shown to be effective on the case study considered, using less

computational time while giving results with accuracy comparable to that of the best approach (Incremental Learning).

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 vectors, in this paper the main novelty lies in the proposed cost-effective update based on the special method of feature vector selection, i.e. 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). Comparing to SOGP, the online approach proposed in this paper cannot bound the data points in the model, but the novel 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.

REFERENCES

G. Baudat and F. Anouar, "Feature Vector Selection and projection using kernels," Neurocomputing, vol. 55, no. 1-2, pp.21-38, 2003.

A. Bordes, S. Ertekin, J. Weston, and L. Bottou, "Fast kernel classifiers with online and active learning," The Journal of Machine Learning Research, vol. 6, pp.1579-1619, 2005.

Csató, L., & Opper, M. (2002). Sparse on-line Gaussian processes. Neural computation, 14(3), 641-668.

G. Cauwenberghs and T. Poggio, "Incremental and decremental support vector machine learning," Fourteenth conference on advances in neural information processing systems, NIPS, pp. 409-415, 2001.

K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive-aggressive algorithms," The Journal of Machine Learning Research, vol. 7, pp.551-585, 2006.

S. Dai, C. Wang and M. Wang, "Dynamic learning from adaptive neural network control of a class of nonaffine nonlinear systems," IEEE Transaction on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 111-123, 2014.

K. B. Dyer, R. Capo and R. Polikar, "COMPOSE: A semisupervised Learning Framework for intitally labeled nonstationary streaming data," IEEE Transaction on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 12-26, 2014.

Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," Signal Processing, IEEE Transactions on, vol. 52, no. 8, pp.2275-2285, 2004.

J. B. Gao, S. R. Gunn, C. J. Harris, and M. Brown, "A probabilistic framework for SVM regression and error bar estimation," Machine Learning, vol. 46, no. 1-3, pp. 71-89, 2002.

T. Jung, and D. Polani, "Sequential learning with LS-SVM for large-scale data sets," InArtificial Neural Networks–ICANN 2006, pp. 381-390, Springer Berlin Heidelberg, 2006.

M. Karasuyama, and I. Takeuchi, "Multiple incremental decremental learning of support vector machines," IEEE Transactions on Neural Networks, vol. 21, no. 7, pp. 1048-1059, 2010.

J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," IEEE Transactions on signal processing, vol. 52, no. 8, pp. 2165-2176, 2004.

L. I. Kuncheva and W. J. Faithfull, "PCA feature extraction for change detection in multidimensional unlabeled data," IEEE Transaction on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 69-80, 2014.

S. Van Vaerenbergh, M. Lázaro-Gredilla, and I. Santamaría, "Kernel recursive least-squares tracker for time-varying regression". *Neural Networks and Learning Systems, IEEE Transactions on*, vol. *23*, no. 8,

pp.1313-1326, 2012.
J. Liu, R. Seraoui, V. Vitelli, and E. Zio, "Nuclear power plant components condition monitoring by probabilistic support vector machine," Annals of Nuclear Energy, vol. 56, pp. 23-33, 2013).
J. Ma, J. Theiler, S. Perkins, "Accurate on-line support vector regression," Neural Computation, vol. 15, no. 11, pp. 2683-2703, 2003,.
M. Martin, "On-line support vector machine regression." Machine Learning: ECML Springer Berlin Heidelberg, vol. 2002, pp. 282-294, 2002.
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, vol. 26, no. 3, pp. 1960-1971, 2011.
F. Orabona, J. Keshet, and B. Caputo, "Bounded kernel-based online learning," The Journal of Machine Learning Research, vol. 10, pp.2643-2666, 2009.
R. Polikar and C. Alippi, "Guest Editorial Learning in Nonstationary and Evolving Environments," IEEE Trans. Neural Networ. & Learn. Syst., vol. 25, no. 1, pp. 9-11, 2014.
M. Pratama, S. G. Anavatti, P. P. Angelov and E. Lughofer, "PANFIS: A novel incremental learning machine," IEEE Transaction on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 56-68, 2014.
V. Reppa, M. M. Polycarpou and C. G. Panayiotou, "Adaptive approximation for multiple sensor fault detection and isolation of nonlinear uncertain systems," IEEE Transaction on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 137-153, 2014.
D. Wang, B. Zhang, P. Zhang, and H. Qiao, "An online core vector machine with adaptive MEB adjustment," Pattern Recognition, vo. 43, no. 10, pp. 3468-3482, 2010.
E. Zio, "Diagnostics and prognostics of engineering systems: methods and techniques," Chapter 17. Engineering Science Reference, USA, 2012.
F. Cucker and D. X. Zhou. Learning Theory: "An Approximation Theory Viewpoint," Cambridge University Press, New York, NY, USA, 2007.

PAPER V: JIE LIU & ENRICO ZIO "AN ONLINE LEARNING APPROACH FOR KERNEL-BASED ENSEMBLES WITH DRIFTING DATA STREAM," *IEEE TRANSACTIONS ON NEURAL NETOWORKS AND LEARNING SYSTEMS.* (UNDER REVIEW)

AN ONLINE LEARNING APPROACH FOR KERNEL-BASED ENSEMBLES WITH DRIFTING DATA STREAM

Jie LIU and Enrico ZIO

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. Online learning ensemble is an effective way to tackle this problem and several approaches have already been proposed, including data chunk-based, drift detector-based and instance-based approaches. Data chunk-based and drift detector-based approaches build adaptively new sub-models with a sufficient number of new data points. These approaches can reduce the computational complexity but suffer a delay of updating, compared to instance-based approaches, while updating the ensemble with each new data point in instance-based approaches is time-consuming. In this paper, an instance-based online learning approach is proposed for kernel-based ensembles, which reduces the computational complexity during updating and can follow timely the ongoing patterns by resorting to Feature Vector Selection (FVS). The proposed approach can also create 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. An experiment is carried out on a real case study concerning a component of a Nuclear Power Plant (NPP). Comparisons with several benchmark approaches prove the efficiency and accuracy of the online learning ensemble approach proposed.

Key words: Online learning ensemble, Feature selection, Dynamic ensemble selection, Pattern drifts, Kernel methods, Dynamic weighted ensemble

1. INTRODUCTION

Building an efficient and accurate predictor from available data is one of the main objectives in machine learning. Most of the current approaches are applied only in static environments, i.e. the data are generated from a distribution which does not change with time. On the other hand, in practical applications, the underlying, normally unknown distribution generating the data can vary with time, causing pattern drifts. Pattern drifts can be due to a natural evolution of the environment, changes in the operational conditions or faults affecting the physical system [1]. In such cases, the data are generated from nonstationary environments and models 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 problem, which can be categorized into adaptive single model [2], [3], [4] and online learning ensembles [5], [6], [7]. The former approach is based on an adaptive model that learns incrementally the new patterns and/or forgets the old inefficient ones; 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 approach of online learning ensembles.

There are different types of approaches for online learning ensembles, e.g. data-chunk-based approaches, drift detector-based approaches, instance-based approaches, etc. Accuracy Weighted Ensemble (AWE) is proposed in [8] to train a new classifier on each incoming data chunk and to update the ensemble sub-models weights according to their accuracy on the past and present data chunks. Streaming Ensemble Algorithm [9] builds separate sub-models on sequential data chunks and combines them into a fixed-size ensemble using a heuristic replacement strategy. Learning++.NSE [10] 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. These previous approaches train new sub-models on the new data chunks. Similar approaches are also used in [11], [12] and [13]. 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 or update the ensemble with each single data point. Adaptive Classifier Ensemble (ACE) [14] slowly builds a new sub-model when the sub-models error on the new data reaches a certain threshold. In [15], pattern drifts are detected by measuring the normalized weighted average output of the sub-models in the ensemble. Diversity analysis is used in [16] to divide different drifts. The most popular drift detector algorithm is the Drift Detection Method (DDM) [17], which models the prediction error on each data point according to a binominal distribution. A modified version of DDM, called EDDM is proposed in [18], which gives better results but is more sensitive to noise. A new approach for online learning ensembles, called Diversity for Dealing with Drift (DDD) is proposed in [7], 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 [19], 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. AddExp in [20] adapts models' weights according to their actual losses and a decreasing factor is integrated to reduce the weights of the sub-models which perform poorly. The Incremental Local Learning Soft Sensing Algorithm (ILLSA) [21] is also an instance-based approach, which contains two parts: one is training different sub-models on data points from different patterns; the other part is updating the sub-models weights for each new data point, according to the posterior probability given by a Bayesian framework. Another instance-based approach, named Online Weighted Ensemble (OWE) is proposed in [22] 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 [23] to store only the features most relevant to the learnt concept, which in turn increases the memory efficiency.

Thus, a good approach for online learning ensemble demands for timely updating the ensemble including weights and sub-models, decreasing the computational burden brought by frequent updating operations and dealing with different types of drifts.

In this paper, an instance-based online learning approach for kernel-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 [24]. Kernel-based ensembles are made of sub-models trained with kernel methods. 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), while the other input vectors can be represented by a linear combination of the FVs selected in RKHS. In our previous work [25], an adaptive online learning approach, named Online-SVR-FID has been proposed for a single Support Vector Regression (SVR) model to effectively follow the ongoing patterns by adjusting 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 a single kernel-based 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 single model $M1$ follows always the ongoing patterns. If each single models $M1$ for different periods of the data stream are separately saved as sub-models 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. 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 [26], [27] and [28], 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, an experiment on a real case study concerning the condition of a component of a Nuclear Power Plant (NPP) is carried out. Comparisons with Learn++.NSE and OWE show that the proposed approach gives better results 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 a real case study concerning a large dataset for a component of 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 [24] and [25]. In order to thoroughly explain the process of building and updating an ensemble with OE-FV, FVS [24] and Online-SVR-FID [25] are firstly and briefly reviewed in this section.

2.1 FVS

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 the ones which represent the dimensions of the RKHS related to the dataset as FVs, in order to decrease the complexity of the dataset. The other input vectors in the dataset can be represented by a linear combination of the selected FVs in 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 represent the inner product $\langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \boldsymbol{\varphi}(\boldsymbol{x}_j) \rangle$ in 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 function 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 below:

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

where $K_{S,S}$ is the kernel matrix (gram matrix) of $\mathbf{S}$ and $K_{S,n} = (k(\boldsymbol{x}_i, \boldsymbol{x_n})), i = 1,2, \dots, L$ 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 [24].

The value calculate 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).

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

$$J_S = \sum_{i=1}^{M} J_S(\boldsymbol{x}_i) \quad (4)$$

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

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 are not accurate enough, i.e. $1 - J_S(\boldsymbol{x_n}) < \rho$ & $|\hat{y}_n - y_n| > \theta$ for all sub-models, with $\hat{y}_n$ the predicted value given by one current sub-model and θ is a positive value representing the tolerance on the prediction error.

2.2 Online-SVR-FID

Online-SVR-FID proposed in [25] aims at providing an efficient online learning approach for 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.

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.

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. Thus, 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 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 to decrease the computational burden for instance-based approaches for online learning ensemble. The whole idea is based on the FVS in [24] and Online-SVR-FID in [25]. OE-FV builds automatically an ensemble from a single 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 reoccur, 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. sbu-models trained with kernel ridge regression, SVR, Gaussian process etc.

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

1. Train a model M_1 with kernel methods on the training dataset.
2. Suppose there are *n* sub-models ($M_1, M_2, \ldots, M_n$) in the ensemble when a new data point is coming:
   - 2.1 Calculate the predicted value for the new data 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 the model 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 M_1 is up-to-date;
       - 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;
   - 2.4 Update the prediction error **Er** of each sub-model.

Fig. 1 The main procedure of OE-FV.

3.1 Training of the first sub-model in Ensemble

A single model M_1 is trained on the training dataset which is also the first and basic sub-model in the ensemble (step 1 in Figure 1). In order to reduce the model complexity and computational burden, 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 keeps 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

When a new data point is coming, in order to give a reasonable prediction using the ensemble approach (step 2.1 in Figure 1), we use a dynamic ensemble selection strategy. A dynamic ensemble selection, as presented in [26], [27] and [28], is to select the sub-models that are most relevant to the new data point to calculate their separate prediction, and, 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 [26]. In OE-FV, they are selected by the local fitness of the new data point, calculated by Equation (3), with respect to the FVs in each sub-model. 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). 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 $\hat{y}$ separately the predicted value of selected sub-models and the ensemble.

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

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

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, then, 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

If the local fitness of the new data point with respect to the FVs in each sub-model satisfies the

relation $1 - J_{Si}(\boldsymbol{x}) > \rho$, it is judged as a new FV, and it is added to the first model M_1 that is trained on the training dataset (step 2.2 in Figure 1). The other sub-models are not modified with the new FV, as they represent only the patterns in the data at 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 is increased by one, the model is retrained through 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

Once the new data point is judged as not a new FV, the verification of a changed FV is carried out by calculating the prediction errors (absolute bias between the predicted value and the true output) of all the sub-models. If the prediction errors are all bigger than the preset threshold θ, the new data point is judged as a changed pattern. It is used to replace a 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 an idea for SVR which counts the times of being a support vector in the past SVR models during the adaptive learning process, and 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 its contribution 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 give its similarity with each FV in M_1. A bigger a_i in $\boldsymbol{a}$ represents a larger similarity, thus, a bigger contribution to the prediction of the new data point. Its contribution is 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 2.2.1.1 in Figure 1 proposes to "copy" the model M_1 as a new sub-model and before the replacement, then, 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 represent different steps of the data stream.

If the selected FV in M_1 is not a unique in the ensemble, it is replaced directly by the new data point without adding a new sub-model (step 2.2.1.2).

3.5 Update of the prediction error of sub-models

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 in 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 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 are 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 $\boldsymbol{Er}_{Eoc}$ their prediction errors, β a positive parameter smaller than one and $\hat{\boldsymbol{y}}_i$ is the 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) For M_1, it is different from the above two types of the 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 in 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 in Figure 1. M_1 gives a new prediction for the new data point different from the one calculated in step 2.1 in 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 according to $Er_1 = \beta Er_1 + \left|\hat{y}_{1,new} - y_i\right|$, with $\hat{y}_{1,new}$ is the prediction for the new data point given by updated M_1.

4) If a new sub-model is created during the online learning of the new data point, the prediction error the new sub-model is calculated with $Er_{n+1} = \beta Er_1 + \left|\hat{y}_{1,old} - y_i\right|$, with $\hat{y}_{1,old}$ the prediction for the new data point given by M_1 at step 2.1 in Figure 1 which is not updated yet with the new data point, and Er_1 is 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 model M_1 needs to be updated. However, it is not always possible to find a way to update 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.

Training a classic kennel-based model takes the minimization of the MSE on the training dataset as the objective function. In this paper, M_1 is trained on the FVs and minimizes 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 number of data points considered in the objective function is to minimize the MSE on the data points from i_0 to i. In order to avoid the 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 is 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, OE-FV can timely learn the new patterns compared to data chunk-based and drift detector-based approaches for online learning ensemble. It can instantly follow the pattern

drift in the data, and the online learning ensembles based on data chunk or sliding window can only react after a sufficient number of new data points is available.

The aim of 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.

When a new sub-model need to be created, there is no need to train this new sub-model, as it is a "copy" of the 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 between 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.

The new data points are all used to update the sub-models' weights, and only few of them are used to update the M_1 and create new sub-models. For each new data points, 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 strategies can reduce the computational complexity of the online learning process.

4. REAL CASE STUDY

In this paper, the real case study concerns a time series training dataset from a sensor monitoring the leak flow from the first seal of Reactor Coolant Pump (RCP) in NPP. The normal 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. If RCP fails, the reactor has the risk of melting down, e.g. the disaster in Fukushima after the tsunami. The leaked water is radioactive and may endangers the personal working in the NPP. Thus, the accurate prediction of the leak flow is a very important indicator for the operators.

Figure 2 is the normalized time series dataset which contains 13124 values and are measured every four hours. It is clear that it contains gradual, sudden and recurring data. Suppose the data is $l(t)$, 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 $\boldsymbol{x}(t) = [l(t-9), l(t-8), \dots, l(t)]$.

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 to the ensemble one by one. The

basic models are all built with SVR in this experiment.

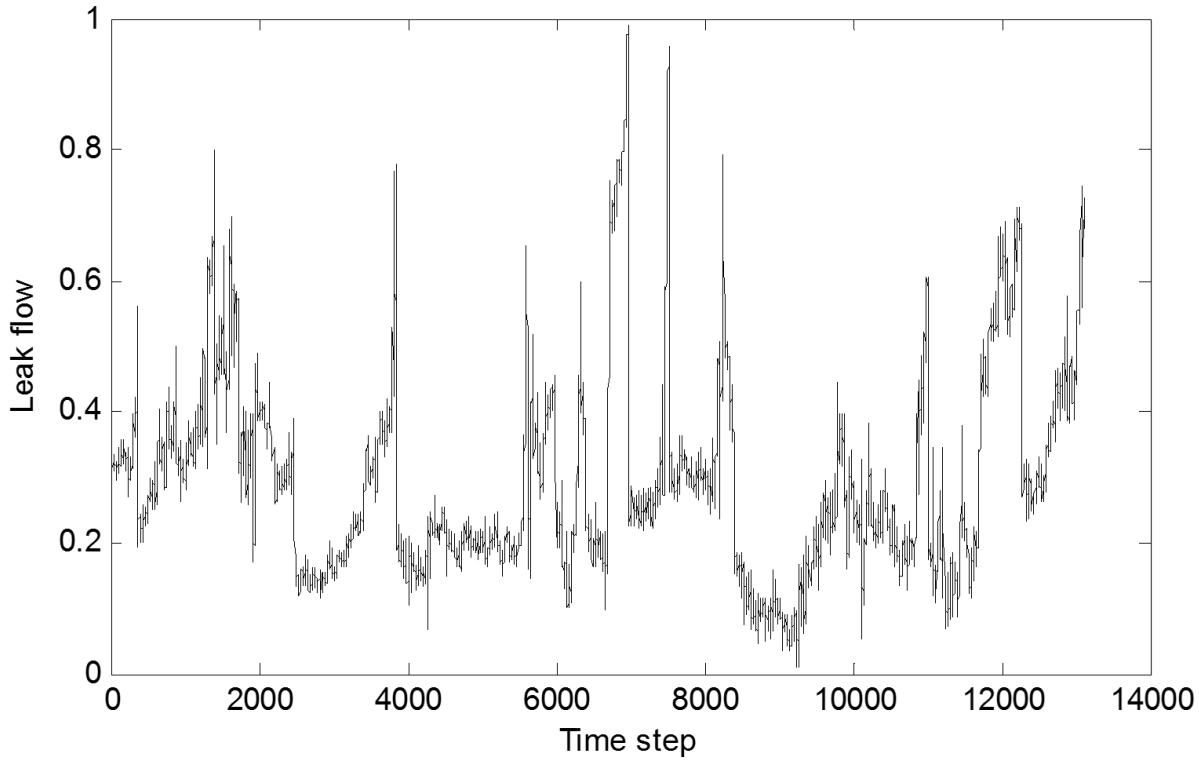

Fig. 2 Data of the leak flow in RCP.

4.1 Prediction results of the proposed ensemble approach

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

The online learning of a single model in [25] and OE-FV, are firstly compared in this experiment.

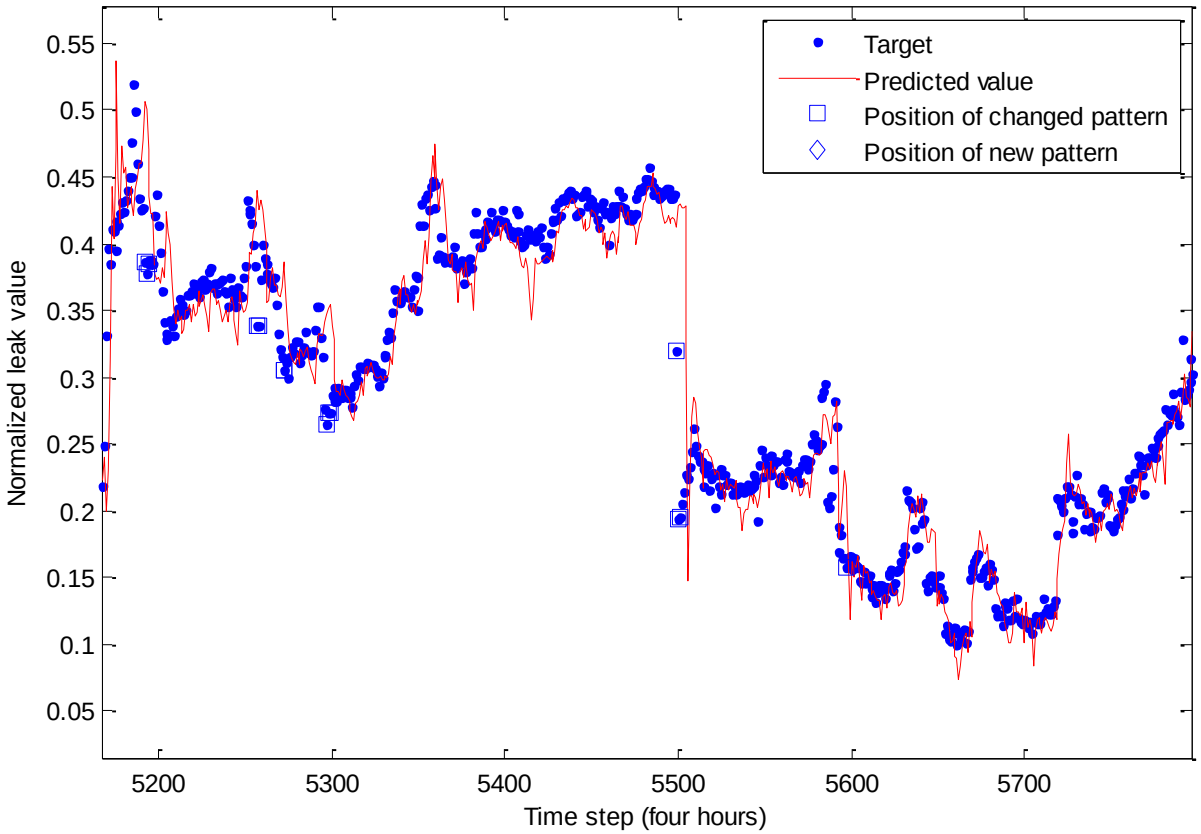

Fig. 3. Prediction results of OE-FV and the positions of changed and new patterns.

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 [25]. In the experiment, there are totally 1198 new data points judged as changed patterns and 13 new data points as new patterns.

While 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, thus the computational complexity, as all the patterns are stored in the ensemble. Thus, OE-FV solves the problem of Online-SVR-FID with recurring patterns. Figure 3 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.

4.2 Results comparisons

In this section, comparisons of experimental results are carried out among Online-SVR-FID [25], Learn++.NSE [10], OWE [22] and the proposed OE-FV, considering the prediction accuracy and the computation time.

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 pattern 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 Learng++.NSE: instead of waiting for a new data chunk, sliding window is integrated. When a new data point is available, the window of a fixed size N moves one step ahead. When the prediction error on the data points in the window exceed 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 also a delay compared to the instance-based approaches for online learning ensemble.

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
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

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

In this case study, by trial-and-error, the size N of the data chunk in Learn++.NSE and of time

window in OWE is fixed at 500. The threshold ε for adding a new sub-model and the discounting rate in calculating prediction error in Learn++.NSE and OWE are separately (0.04, 0.2) and (0.05, 0.3). In the pruned case, the maximal number of sub-models are both 20.

Table I 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.

All these approaches give comparable results considering the prediction accuracy, while Learn++.NSE gives the worst and OE-FV gives 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 that 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 results than data chunk-based or sliding window-based ones in frequently changing environment.

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-model 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 sliding window.

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 reoccur, it has to relearn them before giving a good prediction result. This disadvantage increases the number of updating actions during online learning, thus the computational burden, and decreases the prediction accuracy. While 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 irrelative 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 [22], a larger maximal number of sub-models doesn't always increases the accuracy. The accuracy is no longer improved when the number of sub-models is bigger than a certain value.

Figure 4 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.

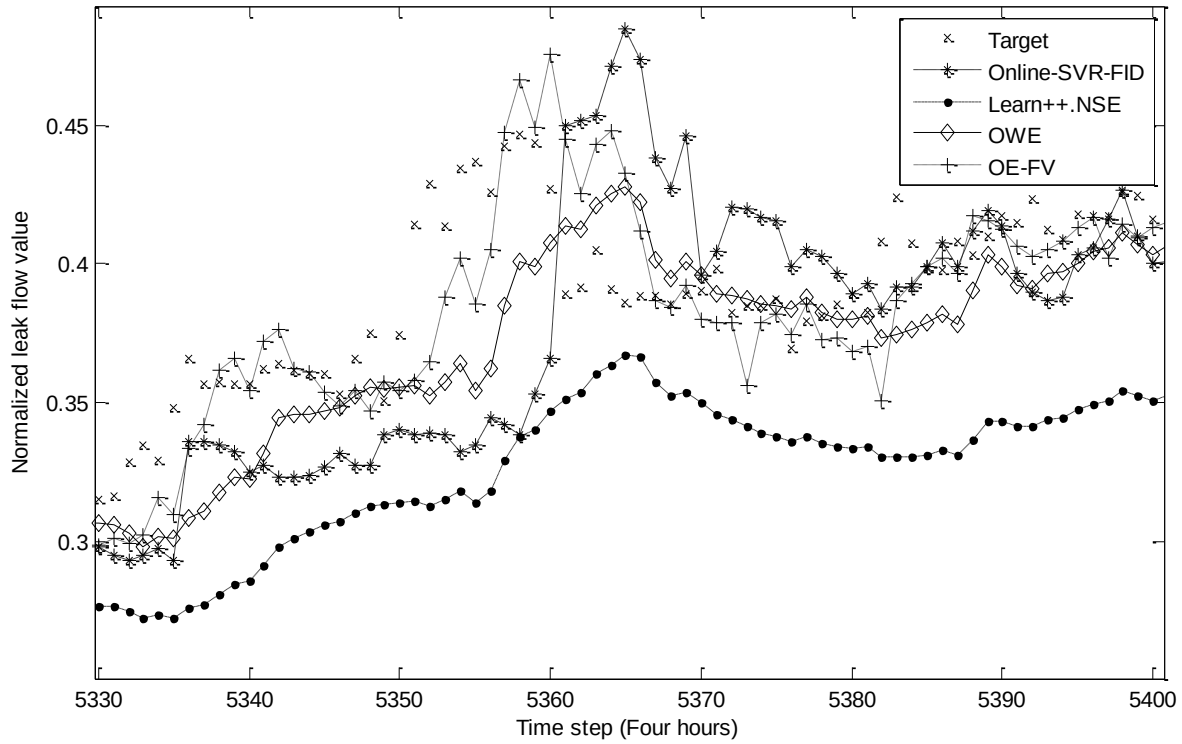

Fig. 4. Comparisons of the prediction results of the test data points from 5300 to 5400.

5. CONCLUSIONS

Based on FVS and Online-SVR-FID, an online learning ensemble approach is proposed for kernel-based models. OE-FV can create an ensemble automatically from a single model. The new sub-models represents separately a certain stage of the first sub-model, whereby the diversity among them is guaranteed. This paradigm is used in online learning ensemble for the first time as the authors know. The dynamic ensemble selection strategy eliminates the irrelevant sub-models 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, dynamic ensemble selection.

Comparisons on a real case study concerning the leak flow in RCP shows that OE-FV outperforms Online-SVR-FID and OWE in both prediction accuracy and computation time. Learn++.NSE uses least time but gives worst prediction results. A drawback of OE-FV is that it is only suitable for kernel-based ensembles.

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] Alippi, Cesare. *Intelligence for Embedded Systems*. Springer, 2014.
[2] S. Dai, C. Wang and M. Wang, “Dynamic learning from adaptive neural network control of a class of nonaffine nonlinear systems,” IEEE Transaction on Neural Networks and Learning Systems, vol. 25, no. 1, pp. 111-123, 2014.
[3] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, “Online passive-aggressive algorithms,” *The Journal of Machine Learning Research*, vol. *7*, pp.551-585, 2006.
[4] Csató, L., & Opper, M. (2002). Sparse on-line Gaussian processes. *Neural computation*, *14*(3), 641-668.
[5] Brzezinski, Dariusz, and Jerzy Stefanowski. "Reacting to different types of concept drift: The accuracy updated ensemble algorithm." *Neural Networks and Learning Systems, IEEE Transactions on* 25.1 (2014): 81-94.
[6] Muhlbaier, Michael D., and Robi Polikar. "An ensemble approach for incremental learning in nonstationary environments." *Multiple classifier systems*. Springer Berlin Heidelberg, 2007. 490-500.
[7] Minku, Leandro L., and Xin Yao. "DDD: A new ensemble approach for dealing with concept drift." *Knowledge and Data Engineering, IEEE Transactions on* 24.4 (2012): 619-633.
[8] Wang, Haixun, et al. "Mining concept-drifting data streams using ensemble classifiers." *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2003.
[9] Street, W. Nick, and YongSeog Kim. "A streaming ensemble algorithm (SEA) for large-scale classification." *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001.
[10] Muhlbaier, Michael D., and Robi Polikar. "An ensemble approach for incremental learning in nonstationary environments." *Multiple classifier systems*. Springer Berlin Heidelberg, 2007. 490-500.
[11] Erdem, Zeki, et al. "Ensemble of SVMs for incremental learning." *Multiple Classifier Systems*. Springer Berlin Heidelberg, 2005. 246-256.
[12] Brzezinski, Dariusz, and Jerzy Stefanowski. "Reacting to different types of concept drift: The accuracy updated ensemble algorithm." *Neural Networks and Learning Systems, IEEE Transactions on* 25.1 (2014): 81-94.
[13] Yang, Xinzhu, Bo Yuan, and Wenhuang Liu. "Dynamic Weighting ensembles for incremental learning." *Proc. of IEEE conference in pattern recognition*. 2009.
[14] Nishida, Kyosuke, Koichiro Yamauchi, and Takashi Omori. "Ace: Adaptive classifiers-ensemble system for concept-drifting environments." *Multiple Classifier Systems*. Springer Berlin Heidelberg, 2005. 176-185.
[15] Razavi-Far, Roozbeh, Piero Baraldi, and Enrico Zio. "Dynamic weighting ensembles for incremental learning and diagnosing new concept class faults in nuclear power systems." *Nuclear Science, IEEE Transactions on* 59.5 (2012): 2520-2530.
[16] Minku, Leandro L., Allan P. White, and Xin Yao. "The impact of diversity on online ensemble learning in the presence of concept drift." *Knowledge and Data Engineering, IEEE Transactions on* 22.5 (2010): 730-742.
[17] Page, E. S. "Continuous inspection schemes." *Biometrika* (1954): 100-115.
[18] Baena-García, Manuel, et al. "Early drift detection method." (2006).
[19] Zliobaite, Indre, et al. "Active learning with drifting streaming data." *IEEE transactions on neural networks and learning systems* 25.1 (2014): 27-39.
[20] Kolter, Jeremy Z., and Marcus A. Maloof. "Using additive expert ensembles to cope with concept drift." *Proceedings of the 22nd international conference on Machine learning*. ACM, 2005.
[21] Kadlec, Petr, and Bogdan Gabrys. "Local learning‐based adaptive soft sensor for catalyst activation prediction." *AIChE Journal* 57.5 (2011): 1288-1301.
[22] Soares, Symone Gomes, and Rui Araújo. "An on-line weighted ensemble of regressor models to handle concept drifts." *Engineering Applications of Artificial Intelligence* 37 (2015): 392-406.
[23] Gomes, João Bártolo, et al. "Mining recurring concepts in a dynamic feature space." (2013): 1-1.

[24] Baudat, Gaston, and Fatiha Anouar. "Feature vector selection and projection using kernels." *Neurocomputing* 55.1-2 (2003): 21-38.
[25] Liu, Jie, and Zio, Enrico. 'An Adaptive Online Learning Approach for Support Vector Regression.' *IEEE Transactions on Neural Networks and Learning Systems*, (submitted).
[26] Ko, Albert HR, Robert Sabourin, and Alceu Souza Britto Jr. "From dynamic classifier selection to dynamic ensemble selection." *Pattern Recognition*41.5 (2008): 1718-1731.
[27] Cavalin, Paulo R., Robert Sabourin, and Ching Y. Suen. "Dynamic selection approaches for multiple classifier systems." *Neural Computing and Applications* 22.3-4 (2013): 673-688.
[28] Cavalin, Paulo R., Robert Sabourin, and Ching Y. Suen. "LoGID: An adaptive framework combining local and global incremental learning for dynamic selection of ensembles of HMMs." *Pattern Recognition* 45.9 (2012): 3544-3556.
[29] Cauwenberghs, Gert, and Tomaso Poggio. "Incremental and decremental support vector machine learning." *Advances in neural information processing systems* (2001): 409-415.

APPENDIX

The pseudo-code of Online-SVR-FID is shown as 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: **S** = [ ]

Threshold of local fitness: ρ

Threshold of prediction error: θ

Offline Training:

First FV in **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 **S** as the first FV.

$\boldsymbol{T}_r$ is reduced as the complement of **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 **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 **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 $1 - J_{\boldsymbol{S},k} \leq \rho$, end the process of FVs selection;

Train the SVR model on the FVs in **S**.

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

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

Selection of the least contribution FV in UPDATE.

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. 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.

PAPER VI: JIE LIU & ENRICO ZIO "REDUCED RANK KERNEL RIDGE REGRESSION-II （RRKRR-II）: A GEOMETRICALLY INTERPRETABLE KERNEL MODEL FOR REGRESSION," *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 2014. (UNDER REVIEW)

REDUCED RANK KERNEL RIDGE REGRESSION-II: A GEOMETRICALLY INTERPRETABLE KERNEL MODEL FOR REGRESSION

Jie Liu, and Enrico Zio, *Senior Member, IEEE*

ABSTRACT

Machine learning methods employing positive kernels have been developed and widely used for classification, regression and unsupervised learning applications, whereby the estimate functions take the form of a weighted sum of kernel expansions. Unacceptable computational burden with large dataset and difficulty in tuning hyperparameters are usually the main drawbacks of kernel methods for large-scale applications. Based on a modified version of the Feature Vector Selection (FVS) method, this paper expresses the estimate function as a weighted sum of the predicted values of the Feature Vectors (FVs), whereby the unknowns in the function are only the predicted values of the FVs. By defining a least square error optimization problem with equal constraints, the analytic solution for these unknowns can be given directly. As an extension of the work of Reduced Rank Kernel Ridge Regression (RRKRR), which applies FVS in Kernel Ridge Regression (KRR), the method here proposed is named RRKRR-II. The tuning of parameters in RRKRR-II is also explained in this paper and shown to be much less complicated than for other kernel methods. Comparisons with some other popular kernel methods for regression on several public datasets show that RRKRR-II gives results comparable with those of the methods which give best results in the experiments in terms of the prediction accuracy with a small subset of the training dataset.

Key words: Computational complexity, Feature vector selection, Kernel method, RRKRR-II, Tuning hyperparameter.

1. INTRODUCTION

In the last decades, benefiting from computational simplicity and good generalization performance in statistical machine learning problems, kernel-based machine learning methods have drawn much attention for regression [25], [26], [23], classification [20], [10], [31] and unsupervised learning [33], [35], [37]. Good and comprehensive reviews of these methods can be found in [27] and [16]. Support Vector Machine (SVM) [1], [34], [6], Kernel Gaussian Process (KGP) [15], [29], [45], Kernel Ridged Regression (KRR) [40], [12], [11], Kernel Logistic Regression (KLR) [51], [19], Kernel Principal Component Analysis (KPCA) [32], [47] are some of the most popular kernel methods for regression.

The nonparametric and semi-parametric representer theorems proposed by Schölkopf *et al.* [36] show that for a large class of kernel algorithms minimizing a sum of an empirical risk term and a regularization term in a Reproducing Kernel Hilbert Space (RKHS), the optimal solutions can be written as a kernel expansions supported on training data points. The estimate function of the kernel methods, including SVM, KGP, KRR, KLR and KPCA, can all be formulated as

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i k(\boldsymbol{x}_i, \boldsymbol{x}) + b, \tag{1}$$

where $f(\boldsymbol{x})$ is the estimate function describing the relation between the data points $(\boldsymbol{x}_i, y_i), i = 1,2, \dots, N$; $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the inwei ner product of the mapping of $\boldsymbol{x}_i, \boldsymbol{x}_j$ in RKHS; α_i are the weights to optimize and b is a constant that can be zero or non-zero. Note that the unknowns in (1) have no practical meanings and that, normally, in kernel methods there are three types of hyperparameters: the penalty factor C which is a trade-off between the empirical risk term and the regularization term, hyperparameters related to the definition of the empirical risk term (e.g. the parameter ϵ in the ϵ-insensitive loss function) and hyperparameters related to the kernel function itself (e.g. the parameter σ in the Gaussian Radial Basis kernel Function (RBF) written as $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\|x\_i - x\_j\|\text{^}2/(2\sigma\text{^}2)}$).

The main drawbacks of standard kernel methods are the unacceptable computational burden for training with large infinite datasets, the difficulty in tuning the hyperparameters and the lack of interpretability of the model.

Many works addressing these drawbacks have been presented in the literature. We focus on SVM for regression and prediction. In order to reduce the computational burden during SVM training, various approaches are proposed to reduce the number of training data points in (1). Some are based on the characteristics of the inputs in RKHS, e.g. KPCA, Feature Vector

Selection (FVS) [2], convex Hull vertices selection [17], Orthogonal Least Squares (OLS) regression [7], Minimum Enclosing Ball (MEB) [42], Sparse Online Gaussian Process (SOGP) [4] etc. Others are based on the prediction accuracy, e.g. orthogonal least squares learning algorithm [8], Fisher Discriminant Analysis [34], significant vector learning [17], kernel F-score feature selection [28], etc. However, these methods all use the same form of the estimate function as in (1), where the weights are just some optimized values without any practical meaning. In [9], Analytic Parameter Selection (APS) is proposed to calculate the hyperparameters values directly from the training dataset. But it is shown that a combination of APS and Genetic Algorithm (GA) can give better prediction results [48]. Many optimization approaches, e.g. Particle Swarm Optimization (PSO) [22], [50], Monte Carlo method (MC) [14], Particle Filtering (PF) [49], Competitive Agglomeration (CA) clustering [18], asymptotically optimal selection [39], are also proposed to find the optimal hyperparameters values. But computational complexity is the main obstacle for these latter approaches for tuning hyperparameters, while APS is easy to realize but it cannot give good results for all hyperparameters, especially the penalty parameter. Finally, so far as the authors know, there have not been any new approaches proposed to tackle the interpretability of an SVM model.

In this paper, by analyzing the distribution property of the inner product (the kernel function is an inner product of two vectors in RKHS) and the geometrical relation between a training data point and the FVs selected with FVS [2], a geometrically interpretable approach is proposed for regression and prediction, which describes the linear relation between the predicted values of FVs and that of any other data point. FVS is used to select the FVs which can represent the dimensions of the training dataset in RKHS, and the linear relation between the predicted value of the FVs and those of the other data points are derived from the general form of the estimate function for kernel methods of (1). In order to keep all the information contained in the selected FVs, an optimization problem with equal constraints (similar to a Least Square-Support Vector Machine (LS-SVM)) is formed to find the minimal Mean Squared Error (MSE) (without regularization term) on the whole training dataset (not only on the selected FVs). Thus, the unknowns in the estimate function of the proposed approach are the predicted values of the FVs and a constant (zero or nonzero), which can be calculated analytically. Minimizing the MSE on the whole training dataset of the model built on the selected FVs can guarantee the generalization performance of the model, even without a regularization term. Equal constraints in the optimization problem keep all the information in the FVs (i.e. no FV is ignored through the loss function, as in SVM) and the optimal values for the unknowns can be calculated

analytically.

The Reduce Rank Kernel Ridge Regression (RRKRR) proposed in [5] integrates FVS in LS-SVM to decrease the size of the training dataset and, thus, the computational complexity. With respect to such work, the proposed approach is named RRKRR-II: the differences between RRKRR and RRKRR-II lie in the objective function used in the optimization and in the estimate function that describes the relation between inputs and outputs. Comparisons on several public datasets show that RRKRR-II always gives better results than RRKRR.

Experiments on several public datasets are carried out. The comparisons with various popular kernel methods considering prediction accuracy and computational burden show that RRKRR-II gives comparable results with the best prediction results of benchmark. According to the experimental results, an efficient method for tuning hyperparameters is given.

The structure of the paper is as follows. Section 2 gives a brief introduction to FVS and the derivation of RRKRR-II is also given in this section, with analytic solutions for the unknowns. Prediction results and comparisons with several popular kernel methods are illustrated in Section 3. Some conclusions and perspectives are drawn in Section 4.

2. RRKRR-II

In this Section, a brief introduction of FVS, proposed in [2], is given with its geometrical interpretation, and RRKRR-II is derived from (1) for kernel methods. An optimization problem is defined to calculate the unknowns in RRKRR-II analytically. Considerations for the optimization problem are also detailed in this Section.

2.1 Feature Vector Selection

FVS, proposed in [2], aims at selecting a number of data points (Feature Vectors (FVs)) $\boldsymbol{S} = (\boldsymbol{x}_i, y_i), i = 1,2,\dots,M$ from the training dataset $\boldsymbol{T} = (\boldsymbol{x}_i, y_i), i = 1,2,\dots,N$, with $M \leq N$, such that the other data points can be expressed as a linear combination of the selected FVs. Suppose $\boldsymbol{\varphi}(\boldsymbol{x})$ is the mapping which maps $\boldsymbol{x}_i$ of each training data point into high dimensional RKHS, and kernel function $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the inner product between two mappings in RKHS, i.e. $\langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \boldsymbol{\varphi}(\boldsymbol{x}_j) \rangle$. For each data point $(\boldsymbol{x}, y)$, its Local Fitness (LF) with respect to feature space $\boldsymbol{S}$ is calculated as:

$$minimum_{\beta_i} \ \delta(\boldsymbol{x}) = \frac{\left\| \boldsymbol{\varphi}(\boldsymbol{x}) - \sum_{i=1}^{M} \beta_i \boldsymbol{\varphi}(\boldsymbol{x}_i) \right\|^2}{\|\boldsymbol{\varphi}(\boldsymbol{x})\|^2}. \quad (2)$$

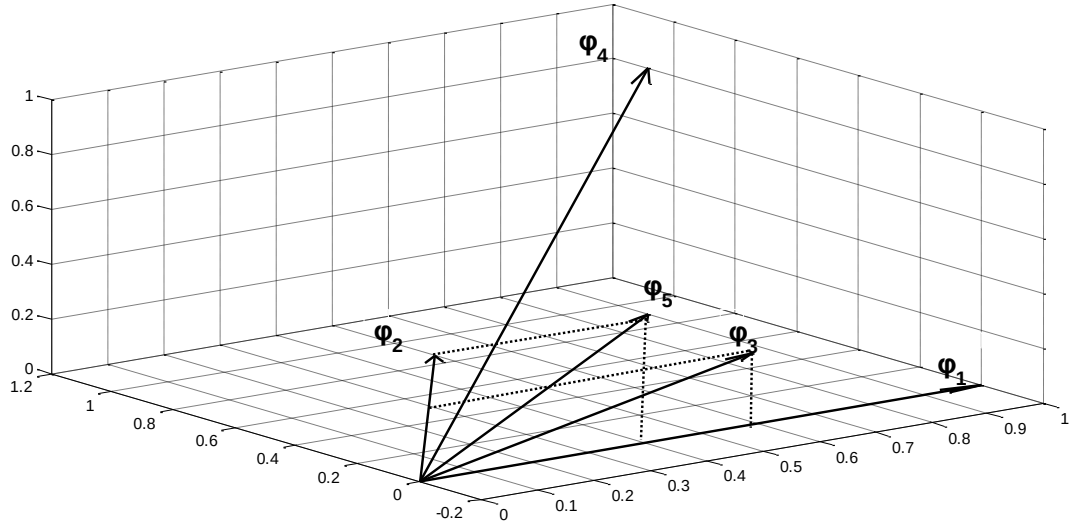

Fig. 1. Geometrical explanation of FVS and $\boldsymbol{\beta}$.

Initialization:
Training dataset: $\mathbf{T} = \{(\boldsymbol{x}_i, y_i)\}$, for $i = 1, 2, \ldots, N$
Feature space: **S** = [ ]
Threshold of LF: τ
FVS:
First FV in **S**:
For i = 1 to N calculate
$\mathbf{S} = \{\boldsymbol{x}_i\}$, compute the sum of the $LF(\boldsymbol{x})$ for all training data points with respect to the present **S**.
End for.
Select the point which gives the minimum of the sum of the $LF(\boldsymbol{x})$ as the first FV and add it to **S** as the first FV.
T is reduced to the complement of **S** in **T** i.e. **T = T\S**.
Second and following FVs:
Calculate local fitness for all data points in **T** with respect to the present feature space **S**;
Select the data point k which gives the maximum of local fitness;
If $LF(\boldsymbol{x}_k) > \tau$, this point is a new FV and added to **S**; and
T = T\E, with **E**=$\{(\boldsymbol{x}_i, y_i)$: $LF(\boldsymbol{x}_i) \leq \tau$ and $(\boldsymbol{x}_k, y_k)\}$;
If $LF(\boldsymbol{x}_k) \leq \tau$, end the process of FVS.

Fig. 2. Pseudo-code of FVS for training dataset $\boldsymbol{T}$

The minimum of $\delta(\boldsymbol{x})$, i.e. LF of $\boldsymbol{x}$, is

$$LF(\boldsymbol{x}) = \left|1 - \frac{K_{S,x}^t K_{S,S}^{-1} K_{S,x}}{k(\boldsymbol{x},\boldsymbol{x})}\right|, \tag{3}$$

where $K_{s,s}$ is the kernel matrix of $\boldsymbol{S}$, and $K_{S,x} = \{k(\boldsymbol{x}_i, \boldsymbol{x})\}, i = 1,2, \ldots, M$. The details of the calculations can be found in the Appendix of [2].

Assuming $\boldsymbol{S}$ is a feature space constructed by only two unit vectors which are linearly independent, but not necessarily orthogonal, i.e. $\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2$ in Fig. 1, any vector $\boldsymbol{\varphi}_3$ in the bi-dimension space of $\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2$, can be expressed as a linear combination of the form $\beta_1\boldsymbol{\varphi}_1 + \beta_2\boldsymbol{\varphi}_2$, with $\beta_1\boldsymbol{\varphi}_1, \beta_2\boldsymbol{\varphi}_2$ being the oblique projections of $\boldsymbol{\varphi}_3$ on $\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2$, respectively, as the dashed lines shown in Fig. 1. For any vector $\boldsymbol{\varphi}_4$ outside the bi-dimension space of $\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2$, the closest vector in the feature space is the projection of $\boldsymbol{\varphi}_4$ on the feature space, i.e. $\boldsymbol{\varphi}_5$ in Fig. 1, and $\beta_1\boldsymbol{\varphi}_1, \beta_2\boldsymbol{\varphi}_2$ are the oblique projections of $\boldsymbol{\varphi}_5$ on $\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2$, with β_1, β_2 calculated by (4). In this case, $\boldsymbol{\varphi}_4$ is a new FV, which extends $\boldsymbol{S}$ to a tri-dimensional feature space.

Thus, geometrically, $LF(\boldsymbol{x}) = \sin^2(\theta)$, with θ the angle between $\boldsymbol{x}$ and feature space $\boldsymbol{S}$.

If $LF(\boldsymbol{x}) = 0$, $\boldsymbol{\varphi}(\boldsymbol{x})$ can be expressed as linear combination of the mapping of FVs, i.e. $\sum_{i=1}^{M} \beta_i \boldsymbol{\varphi}(\boldsymbol{x}_i)$, with $\boldsymbol{\beta} = \{\beta_i\}, i = 1,2,\dots,M$ calculated with the following equation:

$$\boldsymbol{\beta} = K_{S,x}^{t} K_{S,S}^{-1}. \tag{4}$$

If $LF(\boldsymbol{x}) > 0$, $\sum_{i=1}^{M} \beta_i \boldsymbol{\varphi}(\boldsymbol{x}_i)$ with $\boldsymbol{\beta}$ calculated with (4) cannot fully represent $\boldsymbol{\varphi}(\boldsymbol{x})$, i.e. $(\boldsymbol{x}, y)$ is a new FV (pattern).

The pseudo-code of using FVS for selecting FVs in the training dataset $\boldsymbol{T}$ is given in Fig. 2. The process is different from the former works using FVS which is less time-consuming. The most time-consuming part of FVS is the calculation of the local fitness of each data points with respect to the current feature space. In order to reduce the computational complexity of FVS, in this paper, at the end of each iteration for selection of a new FV, the data points in the dataset $\boldsymbol{T}$ which can not be new FV are eliminated and the dataset $\boldsymbol{T}$ is reduced. The local fitness of one data point is smaller with respect to a larger feature space, i.e. its local fitness at the next iteration is smaller than or at least equal to that in the previous iteration. Thus, the data points with the local fitness that satisfies $LF(\boldsymbol{x}_k) \leq \tau$ can not be FVs in the following iteration, as the new FV selected at the next iteration should satisfies $LF(\boldsymbol{x}_k) > \tau$.

In the pseudo-code, a sparsity parameter τ is added as the criterion for new FV selection. After the first FV is selected, the LFs for all data points in $\boldsymbol{T}$ are calculated and the one which gives the maximal LF, e.g. $\boldsymbol{x}_k$ is selected as next possible FV. If $LF(\boldsymbol{x}_k) > \tau$, it is judged as a new FV and added to the present feature space $\boldsymbol{S}$. This process is repeated until the maximal LF with respect to the present feature space $\boldsymbol{S}$ is smaller than or equal to τ. The parameter τ introduced in FVS is to control the sparsity of selected FVs and plays a similar role to ϵ of the ϵ-insensitive loss function in SVR. Geometrically, if a new vector in the feature space is not

contained in the space formed by the present FVs, it is a new FV only if the angle θ between this new vector and the present feature space is bigger than $\arcsin(\sqrt{\tau})$.

2.2 RRKRR-II

Suppose $\boldsymbol{S} = (\boldsymbol{x}_i, y_i), i = 1,2,\dots,M$ are the FVs selected with FVS from the training dataset $\boldsymbol{T}$; for any data point $(\boldsymbol{x}, y)$, its mapping $\boldsymbol{\varphi}(\boldsymbol{x})$ in RKHS can be expressed as a linear combination of the mapping of the selected FVs, i.e. $\sum_{i=1}^{M} \beta_i \boldsymbol{\varphi}(\boldsymbol{x}_i)$ and $\beta_j, j = 1,2,\dots,M$ are multipliers calculated with (4). Note that $f(\boldsymbol{x})$ in (1) can also be written as $f(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i \langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \boldsymbol{\varphi}(\boldsymbol{x}) \rangle + b$, and it can be rewritten as:

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i \langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \sum_{j=1}^{M} \beta_j \boldsymbol{\varphi}(\boldsymbol{x}_j) \rangle + b. \quad (5)$$

By the mathematical distribution property of the inner product, Equation (5) equals to

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \sum_{j=1}^{M} \alpha_i \beta_j \langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \boldsymbol{\varphi}(\boldsymbol{x}_j) \rangle + b$$

$$= \sum_{j=1}^{M} \beta_j (\sum_{i=1}^{N} \alpha_i \langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \boldsymbol{\varphi}(\boldsymbol{x}_j) \rangle) + b. \quad (6)$$

Equation (6) can be further written as

$$f(\boldsymbol{x}) = \sum_{j=1}^{M} \beta_j (f(\boldsymbol{x}_j) - b) + b, \quad (7)$$

where $f(\boldsymbol{x}_j), j = 1,2,\dots,M$ are the predicted values of the FVs in $\boldsymbol{S}$, b is a constant variable and $\beta_j, j = 1,2,\dots,M$ are the multipliers calculated with (4).

Now, the new form of the estimate function of (1) can be written as

$$g(\boldsymbol{x}) = \sum_{j=1}^{M} \beta_j(\boldsymbol{x})(\hat{y}_j - b) + b, \quad (8)$$

where $\hat{y}_j, j = 1,2,\dots,M$ are the predicted values of the FVs selected from the training dataset $\boldsymbol{T}$, b is a constant value, $g(\boldsymbol{x})$ is the prediction of any data point $(\boldsymbol{x}, y)$ and $\beta_j(\boldsymbol{x})$ is the j-th value of the vector $\boldsymbol{\beta}$ calculated by (4), which is dependent only on the input $\boldsymbol{x}$ once the FVs are selected.

Equation (8) describes the linear relation between the predicted values of FVs, i.e. $\hat{y}_j$ and that of any other data point, i.e. $g(\boldsymbol{x})$. This new prediction model is called RRKRR-II. Equation (8) shows that if we know the predicted values for the FVs and the constant b, we can give directly the predicted value for any data point. In the next sub-section, the analytic solutions for the unknowns in (8), i.e. $\hat{y}_j, j = 1,2,\dots,M$ and b are given.

2.3 Optimization Problem Associated to RRKRR-II

In order to calculate the unknowns in (8), an optimization problem is posed, as in all kernel methods. The optimization problem for RRKRR-II is defined as

$$\begin{aligned} &minimize_{\hat{y}_j,b} \qquad W = \frac{1}{N}\sum_{i=1}^{N}(g(\boldsymbol{x_i}) - y_i)^2 \\ &subject\ to \qquad g(\boldsymbol{x_i}) = \sum_{j=1}^{M}\beta_j(\boldsymbol{x_i})(\hat{y}_j - b) + b, \end{aligned} \tag{9}$$

with M representing the number of FVs selected from the whole training dataset $\boldsymbol{T} = (\boldsymbol{x}_i, y_i), i = 1,2,\dots,N$. The optimization problem is trying to find the minimal MSE on the whole training dataset $\boldsymbol{T}$.

Two main challenges of the optimization problem associated to kernel methods are to keep the generalization ability of the model and reduce the computational complexity without losing in prediction accuracy. In RRKRR-II, the FVS procedure selects a small part of the whole training dataset as FVs which are used to build the model, so that the computational complexity is reduced during training and prediction. The objective function in RRKRR-II is the minimal MSE on the whole training dataset, which assures the accuracy of the model on the training dataset and reduces the risk of over-fitting at the same time, as the RRKRR-II model uses only the selected FVs and minimizes the MSE on a much larger dataset.

The regularization term, which is popular in SVM, RRKRR, etc., is not used in RRKRR-II, and experimental results show that the prediction accuracy and generalization ability of RRKRR-II are not decreased compared to other kernel methods. In order not to lose any information contained in the selected FVs, equal constraints are used as in LS-SVM. Thus, two of the three types of hyperparameters in standard kernel methods introduced in the Introduction disappear, i.e. only the hyperparameters related to the kernel function remain in RRKRR-II. On the other hand, an additional parameter, the threshold for new FV selection, i.e. τ, needs to be determined before the training.

After replacing $g(\boldsymbol{x_i})$ in the objective function in (9) with $\sum_{j=1}^{M}\beta_j(\boldsymbol{x_i})(\hat{y}_j - b) + b$, the objective function becomes

$$W = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=1}^{M}\beta_j(\boldsymbol{x_i})\hat{y}_j + b\left(1 - \sum_{j=1}^{M}\beta_j(\boldsymbol{x_i})\right) - y_i\right)^2. \tag{10}$$

Setting the partial derivatives of W with respect to $\hat{y}_j$ and b to zero yields:

$$\frac{\partial W}{\partial \hat{y}_{j_0}} = \sum_{j=1}^{M}\sum_{i=1}^{N}\beta_{j_0}(\boldsymbol{x_i}) * \beta_j(\boldsymbol{x_i}) * \hat{y}_j + b * \sum_{i=1}^{N}\beta_{j_0}(\boldsymbol{x_i}) * \left(1 - \sum_{j=1}^{M}\beta_j(\boldsymbol{x_i})\right) - \sum_{i=1}^{N}\beta_{j_0}(\boldsymbol{x_i}) * y_i = 0 \quad (11)$$

$$\frac{\partial W}{\partial b} = \sum_{j=1}^{M}\sum_{i=1}^{N}\beta_j(\boldsymbol{x}_i) * (1 - \sum_{l=1}^{M}\beta_l(\boldsymbol{x}_i)) * \hat{y}_j + b * \sum_{i=1}^{N}\left(1 - \sum_{j=1}^{M}\beta_j(\boldsymbol{x}_i)\right)^2 - \sum_{i=1}^{N}\left(1 - \sum_{j=1}^{M}\beta_j(\boldsymbol{x}_i)\right) * y_i = 0. \quad (12)$$

These previous Equations (11) and (12) can be expressed as a system of equations as

$$\begin{bmatrix} \mathbf{\Omega} & \mathbf{H} \\ \mathbf{\Gamma}^T & c \end{bmatrix}\begin{bmatrix} \hat{\boldsymbol{y}} \\ b \end{bmatrix} = \begin{bmatrix} \mathbf{P} \\ l \end{bmatrix}, \quad (13)$$

where $\mathbf{\Omega}$ is a $M \times M$ matrix with $\mathbf{\Omega}_{mn} = \sum_{i=1}^{N}\beta_m(\boldsymbol{x_i}) * \beta_n(\boldsymbol{x_i})$, $\mathbf{H}$ is a $M \times 1$ vector with $\mathbf{H}_m = \sum_{i=1}^{N}\beta_m(\boldsymbol{x_i}) * \left(1 - \sum_{j=1}^{M}\beta_j(\boldsymbol{x_i})\right)$, $\mathbf{\Gamma}$ is a $M \times 1$ vector with $\mathbf{\Gamma}_m = \sum_{i=1}^{N}\beta_m(\boldsymbol{x}_i) * (1 - \sum_{l=1}^{M}\beta_l(\boldsymbol{x}_i))$, c is a constant and $c = \sum_{i=1}^{N}\left(1 - \sum_{j=1}^{M}\beta_j(\boldsymbol{x}_i)\right)^2$; $\hat{\boldsymbol{y}} = \left(\hat{y}_j\right), j = 1,2,\dots,M$ and b are the unknowns in (8), $\mathbf{P}$ is a $M \times 1$ vector with $\mathbf{P}_m = \sum_{i=1}^{N}\beta_m(\boldsymbol{x_i}) * y_i$, $l = \sum_{i=1}^{N}\left(1 - \sum_{j=1}^{M}\beta_j(\boldsymbol{x}_i)\right) * y_i$.

TABLE III

MRE on the test dataset

	RRKRR-II	KGP	KPLS	KRR	RRKRR	RVM	SVR
Airfoil	0.06632	0.55447	0.29574	0.03018	0.04133	0.16687	**0.02817**
CCPP	0.15892	0.57247	0.29736	0.03113	**0.02480**	0.22001	0.02523
EMC	0.49079	3.00406	0.31123	**0.02434**	0.45196	0.16248	0.15392
Protein	0.05790	0.81310	0.42075	0.09544	0.06584	NAN	**0.00738**
SARCOS	0.47710	1.08675	0.34493	0.03877	0.03911	0.25101	**0.02284**

The values of the unknowns in (8) can be directly calculated by solving (13).

3. EXPERIMENTAL RESULTS

In this Section, experiments on five public datasets and comparisons with several popular kernel methods are presented to show the performance (prediction accuracy and computational burden) of RRKRR-II. The five public datasets are Airfoil-self-noise dataset (Airfoil) [3], [24]

TABLE I

MSE on the test dataset

	RRKRR-II	KGP	KPLS	KRR	RRKRR	RVM	SVR
Airfoil	0.01456	**0.01349**	0.01568	0.01986	0.02525	0.06169	0.01566
CCPP	0.00189	0.00189	0.00218	0.00187	0.033117	0.01112	**0.00183**
EMC	0.00648	0.00654	0.00701	**0.00631**	0.00962	0.00696	0.00883
Protein	3.46e-07	10370.88	3.29e-07	**2.31e-07**	0.01382	NAN	4.08e-06
SARCOS	6.06e-05	0.00174	0.00218	0.00013	0.01641	0.00124	**6.13e-05**

TABLE II

MRE on the test dataset

	RRKRR-II	KGP	KPLS	KRR	RRKRR	RVM	SVR
Airfoil	0.17867	**0.15322**	0.18545	0.19628	0.25110	0.37256	0.17897
CCPP	0.09215	0.09222	0.09887	0.09109	0.38820	0.19976	**0.08830**
EMC	0.24227	**0.24210**	0.28391	0.27928	0.37602	0.27952	0.34527
Protein	0.00090	224.68	0.00102	**0.00061**	0.23927	NAN	0.00210
SARCOS	0.01191	0.04985	0.07599	0.01559	0.25747	0.06597	**0.01161**

(http://archive.ics.uci.edu/ml/datasets/Airfoil+Self-Noise), Combined Cycle Power Plant dataset (CCPP) [43] (http://archive.ics.uci.edu/ml/datasets/Combined+Cycle+Power+Plant/), the dataset for environmental modelling challenge (EMC) [46] (http://theoval.cmp.uea.ac.uk/~gcc/competition/), Physicochemical Properties of Protein Tertiary Structure Dataset (Protein) provided by Prashant Singh Rana (http://archive.ics.uci.edu/ml/datasets/Physicochemical+Properties+of+Protein+Tertiary+Structure) and the SARCOS dataset (SARCOS) [44] (http://www.gaussianprocess.org/gpml/data/). In the experiments of this paper, all the values of the datasets are normalized between 0.1 and 0.9 at the beginning, and, then, the first 1000 data points of each dataset are chosen as the

training dataset and the test dataset is made of the following 500 data points.

The benchmark methods are Kernel Gaussian Process (KGP) [29], Kernel Partial Least Square (KPLS) [30], KRR [40], RRKRR [2], Relevance Vector Machine (RVM) [41] and SVR [38]. In the experiments, KMBOX-0.9 Matlab toolbox (http://sourceforge.net/projects/kmbox/files/) is used to realize the simulation with KRR, Spider Matlab toolbox (http://people.kyb.tuebingen.mpg.de/spider/main.html) is used for simulations with KGP, KPLS, RVM and SVR. The hyperparameters of the different methods are all tuned by grid search method [21], which finds the best combination of the hyperparameters values among a number of given possible values for each hyperparameter.

3.1 Prediction Accuracy

Table. I and Table. II show separately the MSE and Mean Relative Error (MRE) on the test datasets with different kernel regression approaches; the bold values are the best results given by all the regression approaches. All the methods are working well on all datasets, except for the dataset of Protein where RVM does not work well and KGP gives very bad results.

In all these experiments, RRKRR and RRKRR-II use the same FVs selected from the training datasets. According to the two tables above, the proposed method RRKRR-II always gives better results than RRKRR considering the MSE and MRE on the test datasets. It is also shown that the prediction accuracy of RRKRR-II is always comparable with the best prediction results of all benchmark methods for all public datasets. The prediction results show that RRKRR-II is more stable than other kernel methods. The experiments also prove that when we train a RRKRR-II model with selected FVs through minimizing the MSE on the whole training dataset, there is no need to add a regularization term in the objective function in (9), with no loss of prediction accuracy on the test dataset, i.e. the generalization ability of the model is not decreased. Thus, we might draw a more general conclusion on the training of a kernel regression model: in the optimization problem, if the verification dataset (the whole training dataset in RRKRR-II) is much bigger than the training dataset (selected FVs in RRKRR-II), the

TABLE IV

Sparsity of the RRKRR-II model on different datasets

	Airfoil	CCPP	EMC	Protein	SARCOS
# of FVs	11	39	151	9	71

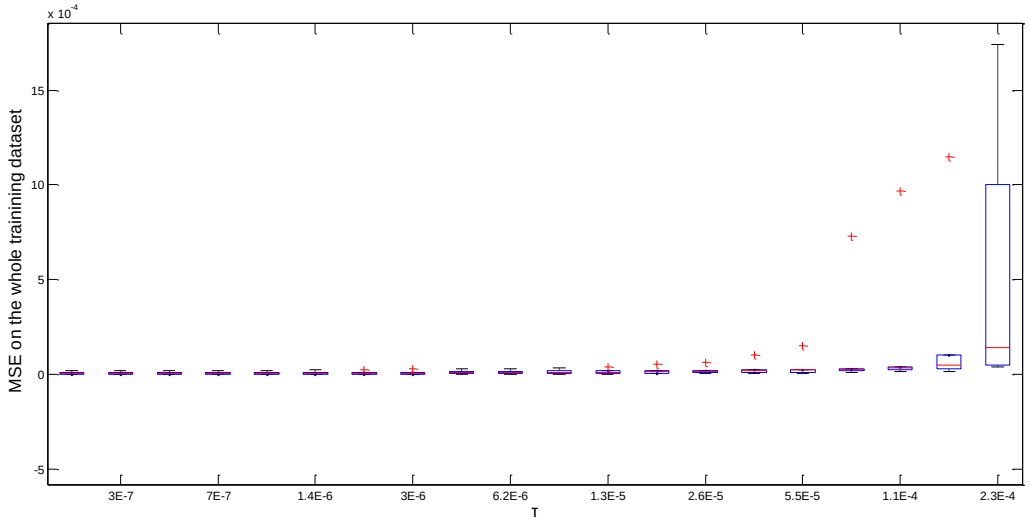

Fig. 3. Boxplot of MSE on the whole training dataset of SARCOS for different values of σ and the same τ.

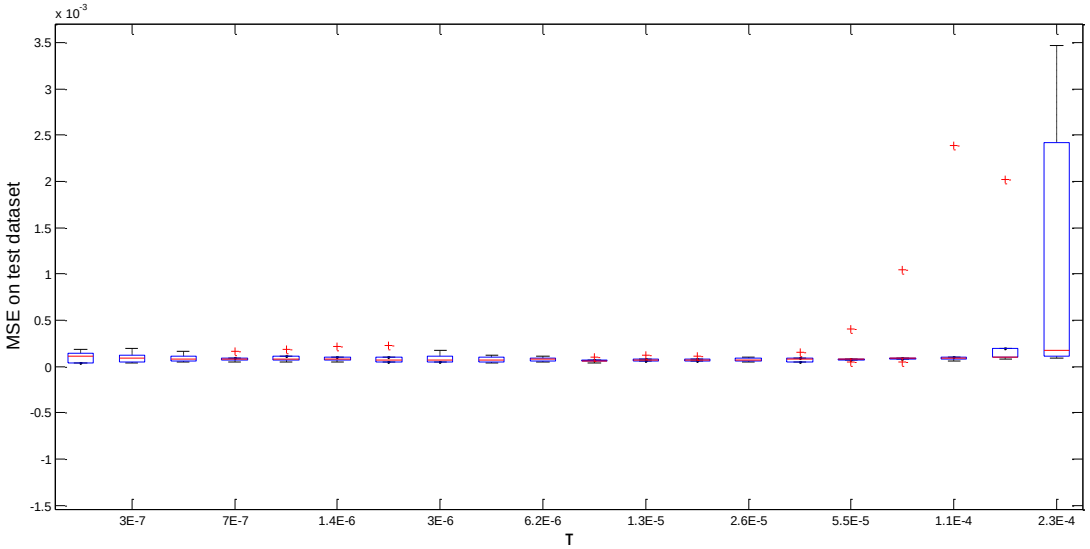

Fig. 4. Boxplot of MSE on the test dataset of SARCOS for different values of σ and the same τ.

regularization term does not play an important role for guaranteeing the generalization ability of the model on the test dataset.

3.2 Computational Burden

Table. III shows the computation time for the prediction of 500 data points in the test datasets

obtained with the same computer (Intel Core i5 CPU 2.5 GHz; RAM 4G); the bold values are the best results given by all the regression approaches. It is shown that RRKRR-II does not take the longest time among all the kernel methods but it still takes much more time than the ones which are the fastest. This is caused by the proposed form of the estimate function. Note that the inverse of $K_{S,S}$ in (4) is not calculated each time for a test data point, and it is stored for use once calculated. Suppose that M data points out of N training points are selected as FVs: the computation complexity of (8) is M^2 and that of the standard one in (1) is N. Thus, the less FVs are selected from the training dataset, e.g. for Airfoil and Protein as shown in Table IV, the faster is the RRKRR-II model for prediction.

As various approaches are proposed for tuning the hyperparameters of the different kernel methods, it is difficult to find a reasonable way to compare the computational burden of the training process (including hyperparameters tuning and solving of the corresponding optimization problem) of the different approaches. However, we still want to point out the advantages of the proposed RRKRR-II approach, although somewhat qualitative. With the FVS process, only a small proportion of training data points are selected, as shown in Table. IV, and thus, the complexity of the model and the computational burden for tuning and optimization is decreased. An upper bound of the number of FVs can be added to the FV selection procedure in Fig. 2 to control the maximal number of FVs. In Fig. 2, the pseudo-code tries to select a subset of FVs which can represent the other data points in RKHS with a maximal difference of τ. In practical applications, in order to decrease the harshness on the tuning of hyperparameters τ and σ, the FV selection can also be stopped when the number of selected FVs reaches the preset upper bound even if the present feature space cannot well represent all the training data points with the tuned values of τ and σ. Compared to classical kernel methods, e.g. KRR, SVR, KGP, the FVS process brings additional computational burden to the training of RRKRR-II, although the FVS process is already simplified compared to the original work of [2]. The added upper bound for the number of FVs can also bound this additional computational burden, as the more FVs are selected, the longer time the FVS process takes.

The tuning of hyperparameters is a big challenge for kernel methods as mentioned in the Introduction. For RRKRR-II, with the disappearance of the regularization term, there are only two parameters that need to be tuned, i.e. τ and σ. Fig. 3 and 4 show separately the boxplot of MSE on the SARCOS training dataset and test dataset for different values of σ (i.e. $\sigma^2 = 0.0013, 0.0029, 0.0084, 0.0211, 0.0529, 0.1330, 0.2685$) and the same τ. From Fig. 3, we

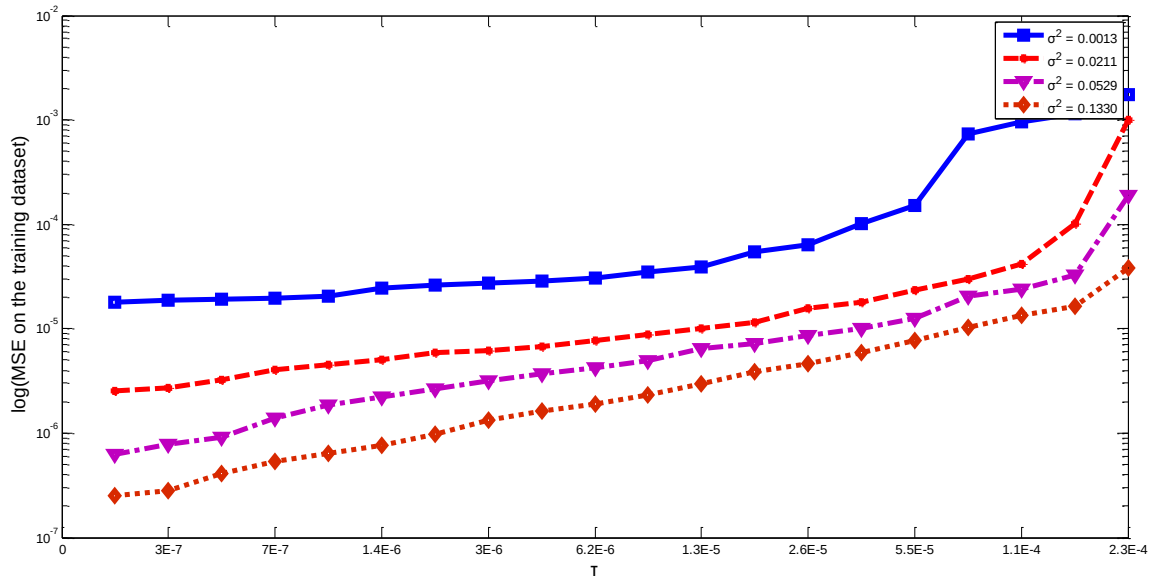

Fig. 5. MSE on the whole training dataset for different values of τ and the same σ^2.

can observe that the prediction accuracy on the training dataset is improved when the value of τ decreases, because more training data points are selected as FVs. When τ is small enough, e.g. smaller than 1.3×10^{-5}, for different values of σ RRKRR-II gives comparable result, and the prediction accuracy on the training data are no longer significantly improved with an even smaller value of τ. Fig. 4 shows that the prediction accuracy for the test dataset is improved at the beginning when the value of τ starts to decrease and different values of σ give comparable results after a certain value for τ, i.e. 1.3×10^{-5}; but when τ is too small, e.g. smaller than 6.2×10^{-6}, the prediction accuracy for the test dataset becomes worse than the case where τ equals 1.3×10^{-5}. This is because the RRKRR-II model has a higher chance of overfitting on the training dataset as more FVs are selected from the training dataset with a smaller τ and the overfitting limits the generalization ability of the model.

In the experiments, the values of σ can be calculated by APS proposed by [9] with (14) below, whereas the value for μ is chosen between 0 and 1. For the experimental results of RRKRR-II shown in Section 3, the value of σ are all calculated with (14) and the values of μ is 0.02 for all public datasets;

$$\sigma^2 = \mu * max\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2, i, j = 1, \dots, N. \tag{14}$$

Once the value for σ is given, we just need to decide the value of τ for FVS. Fig. 5 and 6 show separately the change of MSE on the training and test datasets for different values of τ (i.e. $\tau = 3\times10^{-7}, 7\times10^{-7}, 1.4\times10^{-6}, 3\times10^{-6}, 6.2\times10^{-6}, 1.3\times10^{-5}, 2.6\times10^{-5}, 5.5\times10^{-5}, 1.1\times10^{-4}, 2.3\times10^{-4}$) whereas σ is fixed. For a smaller τ, more FVs are selected and

the prediction accuracy on the training dataset is better, as shown in Fig. 5. But this is not the case for test dataset, as shown in Fig. 6. When the value for τ is smaller than some value (this value is different for different σ and normally it is smaller for a bigger σ), the prediction accuracy on the test dataset becomes worse. The best prediction results for the test dataset are reached when the number of FVs selected from the training dataset is around 70. More selected FVs would cause the overfitting on the training dataset and decrease the generalization ability of the RRKRR-II model. This is why a very small τ increases the prediction accuracy on the training dataset but decreases that on the test dataset. Thus, the value of τ should not be too

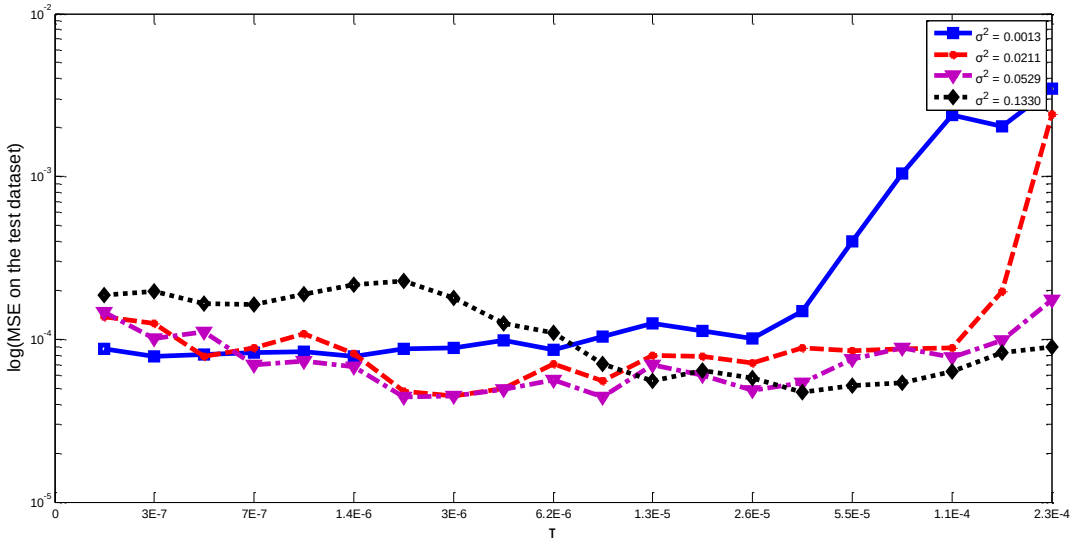

Fig. 6. MSE on the test dataset for different values of τ and the same σ^2.

Fix the value of μ and calculate the value of σ
Initialize the value of τ and the step width $\Delta\tau$
Stop criterion for tuning hyperparameters, π

Calculate MSE on Training dataset with the present values of σ and τ
Repeat
    Decrease the value of τ by $\Delta\tau$ and calculate the MSE on the training dataset
    Calculate the difference of MSE, i.e. ΔMSE
Until ΔMSE < π

Fig. 7. Procedure for tuning hyperparameters of RRKRR-II.

small.

We propose the following process for tuning the value of τ: the value of τ is decreased at a fixed step and when the MSE values on the training dataset between two consequent steps are close enough, the tuning process ends and the value retained is chosen as the best value for τ. The detailed procedure for tuning hyperparameters is shown in Fig. 7.

In addition, RRKRR-II can directly give an analytic solution to the optimization problem like

KRR and RRKRR, which makes the optimization process faster than the searching methods.

Thus, although RRKRR-II may increase the computational burden of the test process, the prediction accuracy and computational complexity associated to the tuning of the hyperparameters and to the model training are better than or comparable with other kernel methods.

4. CONCLUSION

In this paper, a geometrically interpretable kernel approach is proposed through analyzing the geometrical relation between the FVs selected by FVS and the other points in RKHS. The proposed approach is name RRKRR-II, with respect to the former work of Cawley and Talbot (2002). RRKRR-II describes the predicted value of any data point as a weighted sum of the predicted values of the selected FVs. The number of FVs can be bounded by a manually preset upper bound. A simple and efficient strategy is proposed for tuning the two parameters associated to RRKRR-II. Experiments on several public datasets prove that RRKRR-II gives comparable prediction accuracy with the best results given by the benchmark kernel methods. The drawback with RRKRR-II is the additional computational burden brought by the FVS process before training.

Future work will focus on the estimation of the uncertainties associated with the predicted values and on the efficient and adaptive updating of the RRKRR-II model for predictions in dynamic changing environments, considering its geometrical interpretation.

REFERENCES

S. I. Amari, and S. Wu, "Improving support vector machine classifiers by modifying kernel functions," *Neural Networks*, vol. 12, no. 6, pp.783-789, 1999.

G. Baudat and F. Anouar, "Feature Vector Selection and projection using kernels," Neurocomputing, vol. 55, no. 1-2, pp.21-38, 2003.

T. F. Brooks, D. S. Pope, and M. A. Marcolini, "Airfoil self-noise and prediction," (Vol. 1218). *National Aeronautics and Space Administration, Office of Management, Scientific and Technical Information Division*, 1989.

L. Csató, and M. Opper, "Sparse on-line Gaussian processes," *Neural computation*, vol. 14, no. 3, pp.641-668, 2002.

G. C. Cawley, and N. L. Talbot, "Reduced rank kernel ridge regression," *Neural Processing Letters*, vol. 16, no. 3, pp.293-302, 2002.

O. Chapelle, "Training a support vector machine in the primal," *Neural Computation*, vol. 19, no. 5, pp.1155-1178, 2007.

S. Chen, "Local regularization assisted orthogonal least squares regression," *Neurocomputing*, vol. 69, no. 4, pp.559-585, 2006.

S. Chen, C. F. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," Neural Networks, IEEE Transactions on, vol. 2, no. 2, pp.302-309, 1991.

V. Cherkassky, and Y. Ma, "Practical selection of SVM parameters and noise estimation for SVM regression," *Neural networks*, vol. 17, no. 1, pp.113-126,2004.

H. Drucker, S. Wu, and V. N. Vapnik, "Support vector machines for spam categorization," *Neural Networks, IEEE Transactions on*, vol. 10, no. 5, pp.1048-1054, 1999.

G. Fung, and O. L. Mangasarian. "Proximal support vector machine classifiers," *In Proc. of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp.77-86, August, 2001.

T. S. Furey, N. Cristianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Haussler, "Support vector machine classification and validation of cancer tissue samples using microarray expression data," *Bioinformatics*, vol. 16, no. 10, pp.906-914, 2000.

J. Gao, D. Shi, and X. Liu, "Significant vector learning to construct sparse kernel regression models," *Neural Networks*, vol. 20, no. 7, pp.791-798, 2007.

C. Gold, A. Holub, and P. Sollich, "Bayesian approach to feature selection and parameter tuning for support vector machine classifiers," *Neural Networks*, vol. 18, no. 5, pp.693-701, 2005.

T. Hida, "Canonical representations of Gaussian processes and their applications," *Memoirs of the College of Science*, University of Kyoto. Series A: Mathematics, vol. 33, no. 1, pp.109-155, 1960.

T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning," *The annals of statistics*, pp.1171-1220, 2008.

Q. Hong, Z. Bo, and W. Min, "Online support vector machine based on convex hull vertices selection," *IEEE transactions on neural networks and learning systems*, vol. 24, no. 4, pp.593-609, 2013.

J. T. Jeng, "Hybrid approach of selecting hyperparameters of support vector machine for regression," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 36, no. 3, pp.699-709, 2005.

B. Krishnapuram, L. Carin, M. A. Figueiredo, and A. J. Hartemink, "Sparse multinomial logistic regression: Fast algorithms and generalization bounds," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 6, pp.957-968, 2005.

Y. LeCun, L. D. Jackel, L. Bottou, C. Cortes, J. S. Denker, H. Drucker, and V. Vapnik, "Learning algorithms for classification: A comparison on handwritten digit recognition," *Neural networks: the statistical mechanics perspective*, pp.261-276, 1995.

C. J. Lin, C. W. Hsu, and C. C. Chang, "A practical guide to support vector classification. National Taiwan University," www.csie.ntu.edu.tw/cjlin/papers/guide/guide.pdf, 2003.

S. W. Lin, K. C. Ying, S. C. Chen, and Z. J. Lee, "Particle swarm optimization for parameter determination and feature selection of support vector machines," *Expert Systems with Applications*, vol. 35, no. 4, pp.1817-1824, 2008.

J. Liu, R. Seraoui, V. Vitelli, and E. Zio, "Nuclear power plant components condition monitoring by probabilistic support vector machine," *Annals of Nuclear Energy*, vol. 56, pp. 23-33, 2013).

R. Lopez, E. Balsa‐Canto, and E. Onate, "Neural networks for variational problems in engineering," *International Journal for Numerical Methods in Engineering*, vol. 75, no. 11, pp.1341-1360, 2008.

S. Mukherjee, E. Osuna, and F. Girosi, "Nonlinear prediction of chaotic time series using support vector machines," *In Neural Networks for Signal Processing [1997] VII. Proceedings of the 1997 IEEE Workshop*, pp. 511-520, September, 1997.

K. R. Müller, A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik, *Predicting time series with support vector machines. In Artificial Neural Networks—ICANN'97*, Springer Berlin Heidelberg, pp.999-1004, 1997.

K. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf, "An introduction to kernel-based learning algorithms," *Neural Networks, IEEE Transactions on*, vol. 12, no. 2, pp.181-201, 2001.

K. Polat, and S. Güneş," A new feature selection method on classification of medical datasets: Kernel F-score feature selection," *Expert Systems with Applications*, vol. 36, no. 7, pp.10367-10373, 2009.

C. E. Rasmussen, and Z. Ghahramani, "Infinite mixtures of Gaussian process experts," *Advances in neural information processing systems*, vol. 2, pp.881-888, 2002.

R. Rosipal, &and L. J. Trejo, "Kernel partial least squares regression in reproducing kernel hilbert space," *The Journal of Machine Learning Research*, vol. 2, pp.97-123, 2002.

D. Roobaert, and Van M. M. Hulle, "View-based 3d object recognition with support vector machines," *In Neural Networks for Signal Processing IX, 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop*, pp.77-84, August, 1999.

B. Schölkopf, A. Smola, and K. R. Müller, :Kernel principal component analysis," *In Artificial Neural Networks—ICANN'97*, pp.583-588, Springer Berlin Heidelberg, 1997.

B. Schölkopf, A. Smola, and K. R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem. Neural computation," vol. 10, no. 5, pp.1299-1319, 1998.

B. Scholkopft, and K. R. Mullert, "Fisher discriminant analysis with kernels," *Neural networks for signal processing*, vol. IX, 1999.

B. Schölkopf, S. Mika, C. J. Burges, P. Knirsch, K. Muller, G. Ratsch, and A. J. Smola, "Input space versus feature space in kernel-based methods," *Neural Networks, IEEE Transactions on*, vol. 10, no. 5, pp.1000-1017, 1999.

B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem. In Computational learning theory," pp.416-426, Springer Berlin Heidelberg, January, 2001a.

B. Schölkopf, J. C. Platt, , J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp.1443-1471, 2001b.

A. J. Smola, and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp.199-222, 2004.

I. Steinwart, "On the optimal parameter choice for ν-support vector machines," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, no. 10, pp.1274-1284, 2003.

J. A. Suykens , and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp.293-300, 1999.

M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *The journal of machine learning research*, vol. 1, pp.211-244, 2001.

I. W. Tsang, J. T. Kwok, and P. M. Cheung, "Core vector machines: Fast SVM training on very large data sets," *In Journal of Machine Learning Research*, pp.363-392, 2005.

P. Tüfekci, "Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods," *International Journal of Electrical Power & Energy Systems*, vol. 60, pp.126-140, 2014.

S. Vijayakumar, A. D'souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural computation*, vol. 17, no. 12, pp.2602-2634, 2005.

J. Wang, A. Hertzmann, and D. M. Blei, "Gaussian process dynamical models," *In Advances in neural information processing systems*, pp.1441-1448, 2005.

P. Williams, "Using neural networks to model conditional multivariate densities," *Neural Computation*, vol. 8, no. 4, pp.843-854, 1996.

J. Yang, D. Zhang, A. F. Frangi, and J. Y. Yang, "Two-dimensional PCA: a new approach to appearance-based face representation and recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 1, pp.131-137, 2004.

W. Zhao, T. Tao, , and E. Zio, "System reliability prediction by support vector regression with analytic selection and genetic algorithm parameters selection," *Applied Soft Computing*, 2013, (submitted).

W. Zhao, T. Tao, Z. Ding, and E. Zio, "A dynamic particle filter-support vector regression method for reliability prediction," *Reliability Engineering & System Safety*, vol. 119, pp.109-116, 2013b.

W. Zhao, T. Tao, and E. Zio, "A Novel Hybrid Method of Parameters Tuning in Support Vector Regression for Reliability Prediction: Particle Swarm Optimization Combined with Analytical Selection," 2014, (under submission).

J. Zhu, and T. Hastie, "Kernel logistic regression and the import vector machine," *In Advances in neural information processing systems*, pp.1081-1088), 2001.