Procedia Computer Science

Volume 53, 2015, Pages 437–446

2015 INNS Conference on Big Data

Condition Based Maintenance in Railway Transportation Systems Based on Big Data Streaming Analysis

Emanuele Fumeo[1], Luca Oneto[2], and Davide Anguita[1]

[1] DIBRIS, University of Genoa, Via Opera Pia 13, I-16145, Genoa, Italy
(emanuele.fumeo@edu.unige.it, davide.anguita@unige.it)
[2] DITEN, University of Genoa, Via Opera Pia 11A, I-16145, Genoa, Italy (luca.oneto@unige.it)

Abstract

Streaming Data Analysis (SDA) of Big Data Streams (BDS) for Condition Based Maintenance (CBM) in the context of Rail Transportation Systems (RTS) is a state-of-the-art field of research. SDA of BDS is the problem of analyzing, modeling and extracting information from huge amounts of data that continuously come from several sources in real time through computational aware solutions. Among others, CBM for Rail Transportation is one of the most challenging SDA problems, consisting of the implementation of a predictive maintenance system for evaluating the future status of the monitored assets in order to reduce risks related to failures and to avoid service disruptions. The challenge is to collect and analyze all the data streams that come from the numerous on-board sensors monitoring the assets. This paper deals with the problem of CBM applied to the condition monitoring and predictive maintenance of train axle bearings based on sensors data collection, with the purpose of maximizing their Remaining Useful Life (RUL). In particular we propose a novel algorithm for CBM based on SDA that takes advantage of the Online Support Vector Regression (OL-SVR) for predicting the RUL. The novelty of our proposal is the heuristic approach for optimizing the trade-off between the accuracy of the OL-SVR models and the computational time and resources needed in order to build them. Results from tests on a real-world dataset show the actual benefits brought by the proposed methodology.

Keywords: Big Data Streams, Data Analytics, Condition Based Maintenance, Intelligent Transportation Systems, Online Learning, Model Selection

1 Introduction

Today's world is witnessing the exponential growth of data due to the constant generation of new information that has to be collected, stored, and processed [1]. In particular IBM estimates that data in the world is doubling every 18 months [2]. Transforming these large amounts of data into actionable knowledge in a feasible time frame is a key task to map large investments in database storage into an actual advantage for final users [3]. Data analytics algorithms must be able to handle big data by optimizing economic sustainability aspects, which result in resource, time, and accuracy constraints [4].

Selection and peer-review under responsibility of the Scientific Programme Committee of INNS-BigData2015

doi:10.1016/j.procs.2015.07.321

Streaming Data Analysis (SDA) of Big Data Streams (BDS) is becoming a hot research topic [5, 6, 7] since all of the three famous Big Data Vs (Volume, Variety and Velocity) are involved in processing data streams. SDA relates to high volumes of data from different heterogeneous sources, collected in order to effectively extract useful information about systems characterized by a high working speed and affected by noise and other disturbances (high volumes of data do not necessarily bring to higher data quality). Examples of applications are: financial analytics [8] and stock data monitoring [9], intrusion detection and surveillance [10], traffic monitoring [11], real time processing of Radio Frequency Identification readings for events handling [12] and structural health monitoring [13].

Among other fields of applications, Condition Based Maintenance (CBM) is one of the most challenging problems for SDA of BDS [14, 15, 16]. The general principle guiding optimal Maintenance is to make all the necessary actions that guarantee that an asset continues to perform the required functions in the proper way [17, 18]. In other words, the main goal of maintenance is to maximize the lifetime of an asset. Maintenance actions can be framed into a taxonomy, which includes three categories [19]: Corrective Maintenance (CM), Preventive Maintenance (PM), and Condition Based Maintenance (CBM). In CM every time a failure is encountered during the lifetime of an asset, a corrective operation is performed in order to fix the problem, usually by restoring the asset status to normal conditions, or even by substituting a broken part with a new one (sometimes the whole asset). Obviously, this approach is proved to be very expensive, since the costs related to a similar breakdown could easily overcome the asset value. In this case, the combination of direct costs (e.g. failures of connected parts of the system) and indirect costs (e.g. service disruptions) results in an unaffordable situation, consequently confirming that CM is far from being optimal. In PM, instead, reliability measures [20] are considered, such as the Mean Time Before Failure (MTBF), so to perform scheduled maintenance activities along time and to reduce the probability of failures. PM introduces the possibility of planning maintenance actions (replacements, inspections, and so on), so that the time in which the asset is considered unavailable is no more random, instead it is predetermined, except for exceptional events. PM also shows some drawbacks, indeed parts of the assets could be replaced without accurately assessing their real health status, as they get statistically closer and closer to a probable failure. CBM, instead, suggests a prognostic attitude towards maintenance, that can be realized by constantly monitoring the conditions of an asset, consequently allowing triggering maintenance activities only if any potential asset degradation is detected. A further step to this kind of maintenance approach is the possibility to predict possible future issues on the asset by building an in-service degradation model of the asset.

These degradation models can be either based on the physical laws describing the behavior of the asset or data-driven. The latter relies on machine learning algorithms that aim at building models by exploiting data depicting the condition of the monitored asset over its lifetime. In this case, algorithms belonging to the supervised learning framework, where model is inferred from labeled training data, represent a preferable choice. In this framework, Model Selection (MS) addresses the problem of choosing the most suitable model given the available data by properly tuning one or more hyperparameters in order to avoid either under- or overfitting [21]. In order to perform SDA of BDS for CBM algorithms must be able to face two main challenges: (i) to train and select accurate degradation models (i.e. to choose an effective MS strategy); (ii) to deploy such strategy in order to optimize the tradeoff between computational requirements and accuracy. Concerning challenge (i), online training algorithms can help reducing the influence of the training phase on the overall learning process. They have been developed in order to incorporate additional training samples efficiently [22, 23], so that they can deal with BDS. On the contrary, MS procedures have not been adapted to BDS, and traditional ones are often computationally expensive and add a further detrimental effect on challenge (ii), leading to the need of designing the procedure of selecting the best model in a computational efficient way. In other words, the extension to the BDS framework of MS approaches is not as much straightforward, in addition it can be computationally prohibitive, despite representing a desirable feature [22, 24, 25].

In the CBM context, the Remaining Useful Life (RUL) is an important index of the health status of an asset. In order to assess the RUL of an axle bearing, it is necessary to perform a regression analysis over the historical data showing the behavior of the asset from the beginning to the end of its life. In this paper we deal with the problem of estimating the RUL of the train axle bearings based on the data that comes from several on-board sensors. Our proposal takes advantage of the Online Support Vector Regression (OL-SVR) algorithm in order to estimate the RUL of in-service axle bearings and to update the models as soon as new data are available. Every time an axle bearing presents a defect that leads the component to the end of its life (i.e. to failure and to subsequent substitution), we retrieve sensor measurements describing its lifetime and we analyze them. We input data to the OL-SVR algorithm, and we train the associated model with this new data, so to use the generated model as an estimator of the RUL of a train axle bearing. In order to increase the performance of the model, it is extremely important to perform a MS phase, for example by exploiting classical methodologies such as the K-fold Cross Validation (KCV). Unfortunately KCV is too computationally expensive even if the models selected by this method show high accuracy [21]. The proposed solution overtakes these limitations by applying a meta-heuristic optimization approach in order to reduce the time needed to perform this task. The optimization process balances the trade-off between the accuracy that we can achieve with the sub-optimal model we select, and the computational efforts needed in order to find the best model and to assess its performance. In order to prove that our solution is effective, we made use of the Prognostic Health Monitoring (PHM) challenge datasets provided by the FEMTO-ST Institute for the IEEE PHM 2012 Data Challenge [26]. The results show that our solution allows to decrease the computational requirements with respect to complete a MS phase, but improves the accuracy performance of the models with respect to not performing the MS phase.

The rest of the paper is organized as follows. Section 2 describes the problem of CBM for train axle bearings. Section 3 proposed a new computational aware MS strategy for Online Support Vector Regression. Section 4 shows the results of the application of our proposal to the CBM. Finally in Section 5 the conclusions of the paper are drawn.

2 Condition Based Maintenance of Train Axle Bearings

Axle bearings are fundamental rotating elements that aim at constraining relative motion and reducing friction between rotating parts in a machine. In trains, every wheel-axle pair is equipped with an axle bearing. Since high stress and load are applied to these components, they are subjected to deterioration, damages and failures, which have safety and economic implications, that caused a growing scientific and industrial interest about the detection and prediction of axle bearing RUL. This problem is also an example of a challenging real world SDA of BDS problem: in Italy, for example, statistics from the Ministry of Transportation [27] show that, in 2012, every day approximately 8000 trains have traveled along the Italian railway lines. Every train has in average 10 wagons [27], resulting in approximately 80 bearings, a datum that leads to a total number of axle bearings to be monitored in Italy every day approximately equal to 600000. This potentially results in a very large amount of collected data because, according to the literature [28, 29, 30], the average order of magnitude of the frequency of the collection rate suitable for axle bearing condition monitoring is in kHz, since the characteristic vibration frequencies related to bearing damages and defects can be expressed as a function of the dimensions of the axle bearing and of the rotational speed of the considered wheel-axle system. In the Prognostic Health Monitoring (PHM) datasets [26] used to perform the tests described in Section 4, the monitoring system provides data coming from sensors with sampling frequency

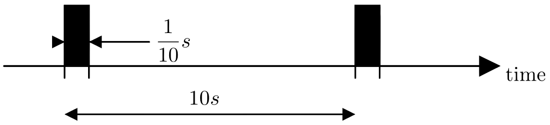

Figure 1: Reading model of the bearing sensors, black blocks are the active reading time of the sensors.

of 25.6 kHz, resulting in approximately 30000 samples every seconds. Measures are taken for 0.1 seconds every 10 seconds (see Figure 1), and they consist of a set of sensor readings, each one including 3 measurements (vertical and horizontal vibration and temperature) for a total of 3 single precision values (12 bytes). The set of information needed to compute the size of data generated daily by the entire monitoring system must be completed with the average in-service time for an Italian train equal to 8 hours, which can be combined with the previously presented information to give a total of approximately 10 TB of data each day. The enormous amount of data generated by this system must be handled and many problems arises, the first of which is to monitor in real time all these bearings. Secondly it is unfeasible to store this kind of data and keep years of history, instead the solution is to compress this information in a model that extracts just the useful one in order to predict the RUL effectively. Lastly it is important to update these models as soon as new data become available by increasing the accuracy of the models and at the same time decreasing the computational effort of performing this task.

3 Online Support Vector Regression: computational-aware models

Let us recall the now-classical regression framework [31] where a set of data $\mathcal{D}_n = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_n, y_n)\}$ is sampled i.i.d. from an unknown distribution μ over $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{X} \in \mathbb{R}^d$ is an input space ($\boldsymbol{x} \in \mathcal{X}$) and $\mathcal{Y} \in \mathbb{R}$ is and output space ($y \in \mathcal{Y}$). The purpose of the learning process is to find a function $f : \mathcal{X} \to \mathcal{Y}$ which best approximates μ. In particular, in the next sections, we will show how to perform this task, when dealing with streaming data, in a computational-aware fashion.

3.1 Online Support Vector Regression

The conventional Support Vector Regression (SVR) [31] searches a regressor in the form $f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b$ which is a linear regressor in the space defined by the nonlinear function $\boldsymbol{\phi} : \mathbb{R}^d \to \mathbb{R}^D$ where usually $D \gg d$. Assuming to use the ϵ-insensitive loss function, $\boldsymbol{w}$ and b are obtained by solving the following Convex Constrained Quadratic Programming (CCQP) optimization problem [31]:

$$\min_{\boldsymbol{w}, b, \boldsymbol{\xi}^+, \boldsymbol{\xi}^-} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C \sum_{i=1}^{n} (\xi_i^+ + \xi_i^-), \quad \text{s.t.} \begin{cases} y_i - \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) - b \leq \epsilon + \xi_i^+, & \forall i \in \{1, \dots, n\} \\ \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b - y_i \leq \epsilon + \xi_i^-, & \forall i \in \{1, \dots, n\} \\ \xi_i^+, \xi_i^- \geq 0, \quad \forall i \in \{1, \dots, n\} \end{cases} \tag{1}$$

where $\epsilon \geq 0$ is the maximum deviation allowed during the training and $C \geq 0$ is the associated penalty for excess deviation during the training. ϵ and C are two hyperparameters that must be aprioristically tuned before solving the CCQP problem. Moreover ξ_i^+ and ξ_i^- correspond to the size of this excess deviation for positive and negative deviations respectively, $\|\boldsymbol{w}\|^2$ is the regularized term, thus it controls the function capacity and finally $\sum_{i=1}^{n}(\xi_i^+ + \xi_i^-)$ is the empirical error measured by the ϵ-insensitive loss function. Since $\boldsymbol{\phi}$ is possibly unknown, it is possible to compute the dual formulation of Problem (1) and obtain [31]:

$$\min_{\boldsymbol{\alpha}^+,\boldsymbol{\alpha}^-} \quad \frac{1}{2}\begin{bmatrix}\boldsymbol{\alpha}^+\\ \boldsymbol{\alpha}^-\end{bmatrix}^T \begin{bmatrix} Q & -Q\\ -Q & Q\end{bmatrix}\begin{bmatrix}\boldsymbol{\alpha}^+\\ \boldsymbol{\alpha}^-\end{bmatrix} + \begin{bmatrix}\boldsymbol{\epsilon}-\boldsymbol{y}\\ \boldsymbol{\epsilon}+\boldsymbol{y}\end{bmatrix}^T\begin{bmatrix}\boldsymbol{\alpha}^+\\ \boldsymbol{\alpha}^-\end{bmatrix}, \quad \text{s.t.} \quad \begin{bmatrix}\mathbf{1}\\ -\mathbf{1}\end{bmatrix}^T\begin{bmatrix}\boldsymbol{\alpha}^+\\ \boldsymbol{\alpha}^-\end{bmatrix} = 0, \quad \mathbf{0} \leq \boldsymbol{\alpha}^+, \boldsymbol{\alpha}^- \leq \boldsymbol{C} \tag{2}$$

where $Q_{ij} = \boldsymbol{\phi}(\boldsymbol{x}_i)^T\boldsymbol{\phi}(\boldsymbol{x}_j)$. By using the kernel trick [32] we do not have to known the $\boldsymbol{\phi}$ explicitly since $K(\boldsymbol{x}_i,\boldsymbol{x}_j) = \boldsymbol{\phi}(\boldsymbol{x}_i)^T\boldsymbol{\phi}(\boldsymbol{x}_j)$. Note that, by knowing $\boldsymbol{\alpha}^+$ and $\boldsymbol{\alpha}^-$ it is possible to reformulate the expression of $f(\boldsymbol{x})$ as $f(\boldsymbol{x}) = \sum_{i=1}^{n}(\alpha_i^+ - \alpha_i^-)K(\boldsymbol{x}_i,\boldsymbol{x}) + b$. One of the most exploited kernel is the Gaussian one, since it enables learning every possible function [33] $K(\boldsymbol{x}_i,\boldsymbol{x}_j) = \exp[-\gamma\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2]$ where γ is a hyperparameter that must be aprioristically tuned (analogously to C) and controls the non-linearity of the regressor. By exploiting the Karush-Kuhn-Tucker (KKT) conditions [31] it is possible to prove that:

$$\left|\begin{array}{c|c|c|c|c} f(\boldsymbol{x}_i) \leq -\epsilon & f(\boldsymbol{x}_i) = -\epsilon & -\epsilon < f(\boldsymbol{x}_i) < \epsilon & f(\boldsymbol{x}_i) = \epsilon & f(\boldsymbol{x}_i) \geq \epsilon \\ \alpha_i^+ = 0 & \alpha_i^+ = 0 & \alpha_i^+ = 0 & 0 \leq \alpha_i^+ \leq C & \alpha_i^+ = C \\ \alpha_i^- = C & 0 \leq \alpha_i^- \leq C & \alpha_i^- = 0 & \alpha_i^- = 0 & \alpha_i^- = 0 \end{array}\right| \tag{3}$$

Note that, the larger ϵ is the faster will be the computation of $f(\boldsymbol{x})$ since most of the α_i^- and α_i^+ will be zero.

In the Online SVM (OL-SVR) approach [23], the regression parameters must be incrementally increased or decreased each time a new sample is added. To achieve this, the five conditions of Eq. (3) can be represented by three subsets into which the samples in a training set $\mathcal{D}_n$ can be classified:

- The $\mathcal{E}$ set. Error support vectors: $\mathcal{E} = \{i : \ |\alpha_i^+ - \alpha_i^-| = C\}$
- The $\mathcal{M}$ set. Margin support vectors: $\mathcal{M} = \{i : \ 0 < |\alpha_i^+ - \alpha_i^-| < C\}$
- The $\mathcal{R}$ set. Remaining samples: $\mathcal{R} = \{i : \ |\alpha_i^+ - \alpha_i^-| = 0\}$

The online algorithm updates the trained SVR function whenever a new sample $(\boldsymbol{x}_{n+1}, y_{n+1})$ is added to (or removed from) the training set $\mathcal{D}_n$. The basic idea is to change the coefficients α_i^+ and α_i^- corresponding to the new sample $\boldsymbol{x}_{n+1}$ in a finite number of discrete steps until it meets the KKT conditions, while ensuring that the existing samples in $\mathcal{D}_n$ continue to satisfy the KKT conditions at each step. Note that, even if we do not go into details with the method for updating the model (it can be retrieved in [23]), the larger ϵ is, the faster will be the update of the model since if the new sample $(\boldsymbol{x}_{n+1}, y_{n+1})$ falls in the ϵ-insensitive tube the update is straightforward: $\alpha_{n+1}^+ = \alpha_{n+1}^- = 0$.

3.2 Online Model Selection

SVR is one of the most effective technique for regression purposes. Unfortunately the SVR learning does not consist just on the training (TR) step where a set of parameters is found by solving the optimization problem described in the previous section. There is also another phase, the Model Selection (MS) phase, where a set of additional variables (hyperparameters) is tuned to find the SVR characterized by optimal performance in classifying previously unseen data [34, 21].

In case of Gaussian kernel-based SVR classifiers, for example, the set of hyperparameters is $\mathcal{H} = \{C, \gamma, \epsilon\}$, where C weights a regularization term, γ tunes the non-linearity of the classifier and ϵ defines the insensitivity tube. In this section we thus focus on the MS step of SVR learning in the OL framework, exploiting an efficient heuristic approach for updating the set of hyperparameters $\mathcal{H}$ when new samples are collected [25, 35]. Typically, in the OL

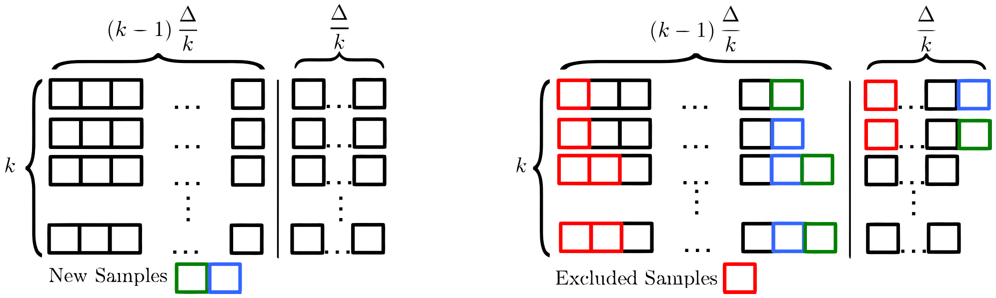

Figure 2: Online k–Fold Cross Validation in the particular case where $\Delta^+ = \Delta^- = \Delta$.

framework, MS is performed only once in a batch mode by exploiting the first n data collected: by using a conventional MS approach, such as the K-Fold Cross Validation (KCV) [36], the SVM hyperparameters are fixed to the optimal value at the n-th step $\mathcal{H}_n^*$ and a first model is trained (f_n). When Δ^+ new samples are gathered, the learning set is updated by adding the most recent samples and, eventually, by discarding Δ^- old ones:

$$\mathcal{D}_{n+\Delta^+-\Delta^-} = \{\mathcal{D}_n \setminus \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_{\Delta^-}, y_{\Delta^-})\}\} \cup \{(\boldsymbol{x}_{n+1}, y_{n+1}), ..., (\boldsymbol{x}_{n+\Delta^+}, y_{n+\Delta^+})\} \tag{4}$$

and $f_{n+\Delta^+-\Delta^-}$ is consequently modified, e.g. accordingly to the method proposed in the previous section. Ideally, we should be able to update both the model and the set of hyperparameters, thanks to the new collected samples. A possible approach consists in performing a full MS-TR procedure at every step, i.e. a complete learning from scratch every time new patterns are collected. However, this approach is too computationally expensive, as described in [25, 35], so an heuristic approach is adopted where a whole re-learning is avoided.

We start by identifying the best hyperparameters set $\mathcal{H}_t^*$ and model f_t at the t-th step with a KCV procedure; we have to keep track of the k models $f_t^{(1)}, \dots, f_t^{(k)}$ trained while applying KCV. When Δ^+ new samples are acquired, we want to modify both the hyperparameters set and the model: it is reasonable to assume that, if Δ^+ and Δ^- are not large (i.e. they are not comparable to n), the hyperparameters will not vary too much from the previous best value. Thus, we can define a neighborhood set $\mathcal{H}_t^\eta$, centered around $\mathcal{H}_t^*$:

$$C \in \left\{C{:}C \in C_{\text{grid}}, C \in \left[\frac{C_t^*}{\eta^C}, \eta^C C_t^*\right]\right\}, \gamma \in \left\{\gamma{:}\gamma \in \gamma_{\text{grid}}, \gamma \in \left[\frac{\gamma_t^*}{\eta^\gamma}, \eta^\gamma \gamma_t^*\right]\right\}, \epsilon \in \left\{\epsilon{:}\epsilon \in \epsilon_{\text{grid}}, \epsilon \in \left[\frac{\epsilon_t^*}{\eta^\epsilon}, \eta^\epsilon \epsilon_t^*\right]\right\} \tag{5}$$

where $\eta^C, \eta^\gamma, \eta^\epsilon \geq 1$ and C_{grid}, γ_{grid} and ϵ_{grid} are predetermined grid of possible values for C, γ and ϵ. We can update the KCV samples set (refer to Fig. 2) and, accordingly, the KCV models $f_t^{(1,\dots,k)}$ for every value of the hyperparameters included in the neighborhood set: through a conventional MS approach based on KCV, we can thus identify the best hyperparameters configuration $\mathcal{H}_{t+1}^* \in \mathcal{H}_t^\eta$. Finally, f_{t+1} is trained accordingly to $\mathcal{H}_t^*$ and the Δ new samples collected by exploiting the procedure described in the previous section. We define this approach as OL-MS. The supplementary computational burden is limited: in fact, we have to update $k+1$ models instead of just one and we have to perform a KCV MS at each updating step, but limited to a restricted neighborhood set. Note that, if the $\mathcal{H}_{t+1}^* \neq \mathcal{H}_t^*$ the neighborhood set of $\mathcal{H}_{t+1}^*$ will be different from the neighborhood set of $\mathcal{H}_t^*$, so we may have to update old models with the most recent data.

3.3 Computational-aware models

In this section we propose a computational-aware heuristics for optimizing all the parameters involved in the procedure described in the previous section. In particular the problem is to find the best value of η^C, η^γ and η^ϵ so to maximize the accuracy of the regressor and minimize the computational effort for achieving this goal. In order to address this issue we use a meta–optimization heuristic technique. Our proposal exploits the simple rule proposed by Baba in [37, 38]. Let us take our set of hyperparameters η^C, η^γ and η^ϵ which defines the neighborhood set and let us optimize their value based on Algorithm 1. Note that $\delta, \eta_0^C, \eta_0^\gamma, \eta_0^\epsilon$ are parameters that balance the tradeoff between accuracy of the final model and computational requirements of our model. Also note that the computational requirements of the complete procedure are basically proportional to the size of the neighborhood set; the larger is the neighborhood, the larger will be the number of models to update with the procedure of Section 3.2. Then we can show some interesting cases:

Algorithm 1: The Computational Aware OL-SVR MS meta-heuristic optimization approach.

```
Require: $\mathcal{D}_n$, $[\eta_0^C, \eta_0^\gamma, \eta_0^\epsilon]$, $\delta$, $C_{\text{grid}}$, $\gamma_{\text{grid}}$ and $\epsilon_{\text{grid}}$

Search $\mathcal{H}_0^*$ with $[\eta_0^C, \eta_0^\gamma, \eta_0^\epsilon]$ as described in Section 3.2
Set $L_{KCV}(f_0)$ as the KCV error for $\mathcal{H}_0^*$
Set a tridimensional vector $\boldsymbol{m} = \mathbf{0}$
for $t \in \{1, 2, \dots\}$ do
    $\Delta^+$ are available and $\Delta^-$ can be disregarded
    Generate a tridimensional Gaussian random vector $\boldsymbol{r}$ centered in $\boldsymbol{m}$
    Set $[\eta_{t+1}^C, \eta_{t+1}^\gamma, \eta_{t+1}^\epsilon] = [\max(1, \eta_t^C + r_1), \max(1, \eta_t^\gamma + r_2), \max(1, \eta_t^\epsilon + r_3)]$
    Find $\mathcal{H}_{t+1}^*$ and $L_{KCV}(f_{t+1})$ with $[\eta_{t+1}^C, \eta_{t+1}^\gamma, \eta_{t+1}^\epsilon]$ as described in Section 3.2
    if $L_{KCV}(f_{t+1}) < L_{KCV}(f_t) + \delta$ then
        $\boldsymbol{m} = 0.4 \cdot \boldsymbol{r} + 0.2 \cdot \boldsymbol{m}$
    else
        $[\eta_{t+1}^C, \eta_{t+1}^\gamma, \eta_{t+1}^\epsilon] = [\eta_t^C - r_1, \eta_t^\gamma - r_2, \eta_t^\epsilon - r_3]$
        $\boldsymbol{m} = 0.5 \cdot \boldsymbol{m}$
    end if
end for
```

- if $\delta = \infty$ and $\eta_0^C = \eta_0^\gamma = \eta_0^\epsilon = 1$ we never move $\mathcal{H}_0^*$ so basically we do not perform any model selection
- if $\delta = 0$ and $\eta_0^C = \eta_0^\gamma = \eta_0^\epsilon \gg 1$ we start with a big neighborhood set and we are willing to shrink it only if the KCV error strictly decreases
- if $\delta \neq 0$ and $\eta_0^C = \eta_0^\gamma = \eta_0^\epsilon > 1$ we have a mixed behavior, we want to shrink the neighborhood set if the KCV error does not change too much

4 Experimental Results

We evaluated the performance of our method on the Prognostic Health Monitoring (PHM) challenge datasets, provided by the FEMTO-ST Institute for the IEEE PHM 2012 Data Challenge [26]. Each dataset includes sensors data collected during a run-to-failure experiment of a single ball bearing using the PRONOSTIA platform, developed by the FEMTO-ST Institute in order to study ball bearings damages and failures. The experiments have been conducted under different load and rotational speed settings so to obtain a different behavior for every bearing. Sensors data is composed of vibration (both in the vertical and the horizontal directions) and temperature measurements, extracted from three signals that are sampled as described in Section 2. The goal of the tests is to assess the performance of the OL-SVR regressor in estimating the RUL of the axle bearings under test. Every axle bearing dataset is treated as the complete

	Parameters	**MAPE (%)**	**Time (s)**	**Speedup**	**Mem (KB)**
C-MS	-	2.57 ± 0.11	18398	$\times 1$	15400
No-MS	-	6.01 ± 0.34	1147	$\times 16$	5
CA-MSS (1)	$\delta = 0, \eta_0^C = \eta_0^\gamma = \eta_0^\epsilon = 100$	3.4 ± 0.17	1901	$\times 10$	474
CA-MSS (2)	$\delta = 0, \eta_0^C = \eta_0^\gamma = \eta_0^\epsilon = 10$	3.5 ± 0.14	1862	$\times 10$	397
CA-MSM (1)	$\delta = 0.1, \eta_0^C = \eta_0^\gamma = \eta_0^\epsilon = 10$	3.1 ± 0.16	1319	$\times 14$	227
CA-MSM (2)	$\delta = 0.3, \eta_0^C = \eta_0^\gamma = \eta_0^\epsilon = 10$	3.2 ± 0.12	1282	$\times 14$	209

Table 1: Prognostic Health Monitoring (PHM) challenge datasets [26]

axle bearing lifetime data coming as a stream from the condition monitoring system connected to that particular bearing. We compare three different approaches to this problem:

- C-MS: Complete MS (no rule, considers the entire parameters sets)
- No-MS: No MS $\Longrightarrow \delta = \infty$ and $\eta_0^C = \eta_0^\gamma = \eta_0^\epsilon = 1$
- CA-MSS: Computational Aware MS Strict approach (the target is to decrease the computational requirements only if the error strictly decreases) $\Longrightarrow \delta = 0$ and $\eta_0^C = \eta_0^\gamma = \eta_0^\epsilon \gg 1$
- CA-MSM: Computational Aware MS Mixed approach (the target is to decrease the computational requirements if the error does not increase too much) $\Longrightarrow \delta \neq 0$ and $\eta_0^C = \eta_0^\gamma = \eta_0^\epsilon > 1$

The hyperparameters sets are defined as $C_{\text{grid}} \in [10^{-4}, 10^2]$, $\gamma_{\text{grid}} \in [10^{-4}, 10^2]$ and $\epsilon_{\text{grid}} \in [0, 10^{-1}]$ sampled in logarithmic scale, 25 points for C, 7 for γ and 4 for ϵ, resulting in a total number of tuples equal to 700. For each tuple we have to keep $k = 4$ models plus one: k for performing the KCV in order to quantify its generalization error, and one, trained with the complete dataset, used as final model of the RUL. We want to record and compare the best models selected by the different approaches based on three different indices of performance:

- Mean Absolute Percentage Error (MAPE) [39] on a set of data that have been hidden from the learning process
- Time requirements (Time) in order to select the model
- Memory requirements (Mem) in order to store all the different models in memory (HDD and RAM)

We perform tests for different values of δ and for different initial conditions over $\mathcal{H}_0 = [\eta_0^C, \eta_0^\gamma, \eta_0^\epsilon]$. Based on the results reported in Table 1, we underline that the C-MS method results to be the best option in terms of accuracy, but the worst in terms of computational requirements. Concerning the No-MS we can derive the opposite conclusions. CA-MSS manages to improve the No-MS in terms of accuracy and the C-MS in terms of computational requirements, but only the CA-MSM is able to take the best of the two worlds and to achieve accuracy results that are very close to the C-MS, as well as a computational burden that is much closer to the No-MS method with respect to CA-MSS.

5 Conclusions

This work focuses on the problem of Streaming Data Analysis of Big Data Streams for Condition Based Maintenance in the context of Rail Transportation Systems. In particular we deal with the problem of predicting the Remaining Useful Life of Train Axle Bearings based on the streams of data that come from the on-board sensors. For this purpose we propose to exploit the Online Support Vector Regression for updating the model as soon as new data become available. Our proposal also consists of a model selection strategy that is able to optimize the tradeoff between accuracy of the final model and resources needed in order to perform the model selection phase itself, an additional desirable feature mostly disregarded due to its computational requirements. Results on real world dataset show the advantages of this solution and prove the generality of the method, which could be successfully used in other applications.

References

[1] V. Mayer-Schönberger, K. Cukier, Big data: A revolution that will transform how we live, work, and think, Houghton Mifflin Harcourt, 2013.

[2] S. Mills, S. Lucas, L. Irakliotis, M. Rappa, T. Carlson, B. Perlowitz, Demystifying big data: a practical guide to transforming the business of government, in: Technical report. http://www. ibm. com/software/data/demystifying-big-data, 2012.

[3] X. Wu, X. Zhu, G. Q. Wu, W. Ding, Data mining with big data, IEEE Transactions on Knowledge and Data Engineering 26 (1) (2014) 97–107.

[4] M. I. Jordan, On statistics, computation and scalability, Bernoulli 19 (4) (2013) 1378–1390.

[5] O. Maimon, L. Rokach, Data mining and knowledge discovery handbook, Springer, 2005.

[6] G. D. G. Morales, A. Bifet, Samoa: Scalable advanced massive online analysis, Journal of Machine Learning Research 16 (2015) 149–153.

[7] B. S. Parker, L. Khan, A. Bifet, Incremental ensemble classifier addressing non-stationary fast data streams, in: IEEE International Conference on Data Mining Workshop, 2014.

[8] M. E. Edge, P. R. F. Sampaio, M. Choudhary, Towards a proactive fraud management framework for financial data streams, in: IEEE International Symposium on Dependable, Autonomic and Secure Computing, 2007.

[9] X. Lian, L. Chen, J. X. Yu, J. Han, J. Ma, Multiscale representations for fast pattern matching in stream time series, IEEE Transactions on Knowledge and Data Engineering 21 (4) (2009) 568–581.

[10] A. H. R. Ko, A. L. Jousselme, P. Maupin, A novel measure for data stream anomaly detection in a bio-surveillance system, in: International Conference on Information Fusion, 2011.

[11] Y. Shiming, K. Kalpakis, A. Biem, Detecting road traffic events by coupling multiple timeseries with a nonparametric bayesian method, IEEE Transactions on Intelligent Transportation Systems 15 (5) (2014) 1936–1946.

[12] E. Wu, Y. Diao, S. Rizvi, High-performance complex event processing over streams, in: ACM SIGMOD international conference on Management of data, 2006.

[13] D. Balageas, C. P. Fritzen, A. Güemes, Structural health monitoring, Wiley Online Library, 2006.

[14] R. K. Youree, J. S. Yalowitz, A. Corder, T. K. Ooi, A multivariate statistical analysis technique for on-line fault prediction, in: International Conference on Prognostics and Health Management, 2008.

[15] W. Sammouri, E. Come, L. Oukhellou, P. Aknin, C. E. Fonlladosa, Floating train data systems for preventive maintenance: A data mining approach, in: International Conference on Industrial Engineering and Systems Management, 2013.

[16] A. Coraddu, L. Oneto, A. Ghio, S. Savio, D. Anguita, M. Figari, Machine learning approaches for improving condition-based maintenance of naval propulsion plants, Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment (in-press) (2014) 1–18.

[17] BSI, Bs 3811-glossary of maintenance management terms in terotechnology (1993).

[18] J. W. Sheppard, M. A. Kaufman, T. J. Wilmer, Ieee standards for prognostics and health management, IEEE Aerospace and Electronic Systems Magazine 24 (9) (2009) 34–41.

[19] G. Budai-Balke, Operations research models for scheduling railway infrastructure maintenance, Rozenberg Publishers, 2009.

[20] E. G. Frankel, Systems reliability and risk analysis, Taylor & Francis, 1988.

[21] D. Anguita, A. Ghio, L. Oneto, S. Ridella, In-sample and out-of-sample model selection and error estimation for support vector machines, IEEE Transactions on Neural Networks and Learning Systems 23 (9) (2012) 1390–1406.

[22] P. Laskov, C. Gehl, S. Krüger, K. R. Müller, Incremental support vector learning: Analysis, implementation and applications, The Journal of Machine Learning Research 7 (2006) 1909–1936.

[23] J. Ma, J. Theiler, S. Perkins, Accurate on-line support vector regression, Neural Computation 15 (11) (2003) 2683–2703.

[24] T. Diethe, M. Girolami, Online learning with (multiple) kernels: A review, Neural computation

25 (3) (2013) 567–625.

[25] D. Anguita, A. Ghio, I. A. Lawal, L. Oneto, A heuristic approach to model selection for online support vector machines, in: International Workshop on Advances in Regularization, Optimization, Kernel Methods and Support Vector Machines: Theory and Application, 2013.

[26] P. Nectoux, R. Gouriveau, K. Medjaher, E. Ramasso, B. Chebel-Morello, N. Zerhouni, C. Varnier, Pronostia: An experimental platform for bearings accelerated degradation tests., in: IEEE International Conference on Prognostics and Health Management, 2012.

[27] U. Statistica, Conto nazionale delle infrastrutture e dei trasporti - anni 2012-2013, in: Ministero Italiano delle Infrastrutture e dei Trasporti, 2012.

[28] R. R. Schoen, T. G. Habetler, F. Kamran, R. G. Bartfield, Motor bearing damage detection using stator current monitoring, IEEE Transactions on Industry Applications 31 (6) (1995) 1274–1279.

[29] S. Nandi, H. A. Toliyat, X. Li, Condition monitoring and fault diagnosis of electrical motors-a review, IEEE Transactions on Energy Conversion 20 (4) (2005) 719–729.

[30] N. Tandon, A. Choudhury, A review of vibration and acoustic measurement methods for the detection of defects in rolling element bearings, Tribology international 32 (8) (1999) 469–480.

[31] B. Schölkopf, A. Smola, Learning with kernels: Support vector machines, regularization, optimization, and beyond, MIT press, 2001.

[32] B. Schölkopf, The kernel trick for distances, in: Neural Information Processing Systems, 2001.

[33] S. S. Keerthi, C. J. Lin, Asymptotic behaviors of support vector machines with gaussian kernel, Neural computation 15 (7) (2003) 1667–1689.

[34] V. N. Vapnik, Statistical learning theory, Wiley-Interscience, 1998.

[35] S. Shalev-Shwartz, Online learning: Theory, algorithms, and applications, Ph.D. thesis, Hebrew University of Jerusalem (2007).

[36] D. Anguita, A. Ghio, S. Ridella, D. Sterpi, K-fold cross validation for error rate estimate in support vector machines., in: DMIN, 2009.

[37] N. Baba, Convergence of a random optimization method for constrained optimization problems, Journal of Optimization Theory and Applications 33 (4) (1981) 451–461.

[38] N. Baba, A new approach for finding the global minimum of error function of neural networks, Neural networks 2 (5) (1989) 367–373.

[39] L. Ghelardoni, A. Ghio, D. Anguita, Energy load forecasting using empirical mode decomposition and support vector regression, IEEE Transactions on Smart Grid 4 (1) (2013) 549–556.