Fault Prognostics by an Ensemble of Echo State Networks in Presence of Event Based Measurements

Mingjing Xu [a], Piero Baraldi [a,*], Sameer Al-Dahidi [b], Enrico Zio [a,c,d,e]

[a] *Energy Department, Politecnico di Milano, Via La Masa 34, 20156 Milan, Italy*
[b] *Mechanical and Maintenance Engineering Department, School of Applied Technical Sciences, German Jordanian University, Amman, Jordan*
[c] *MINES ParisTech, PSL Research University, CRC, Sophia Antipolis, France*
[d] *Eminant Scholar, Department of Nuclear Engineering, College of Engineering, Kyung Hee University, Republic of Korea*
[e] *Aramis Srl, Via pergolesi 5, Milano, Italy*

Abstract

Fault prognostics aims at predicting the degradation of equipment for estimating the Remaining Useful Life (*RUL*). Traditional data-driven fault prognostic approaches face the challenge of dealing with incomplete and noisy data collected at irregular time steps, e.g. in correspondence of the occurrence of triggering events in the system. Since the values of all the signals are missing at the same time and the number of missing data largely exceeds the number of triggering events, missing data reconstruction approaches are difficult to apply. In this context, the objective of the present work is to develop a one-step method, which directly receives in input the event-based measurement and produces in output the system *RUL* with the associated uncertainty. Two strategies based on the use of ensembles of Echo State Networks (*ESNs*), properly adapted to deal with data collected at irregular time steps, have been proposed to this aim. A synthetic and a real-world case study are used to show their effectiveness and their superior performance with respect to state-of-the-art prognostic methods.

Keywords: Prognostics; Missing data; Sliding bearing; Echo State Network; Ensemble; Differential evolution optimization.

Nomenclature

RUL	Remaining Useful Life	*MC*	Memory Capacity
RNNs	Recurrent Neural Network	$\boldsymbol{W}_{in}$	Weight of the connections from the input neurons to the internal neurons
RC	Reservoir Computing	$\boldsymbol{W}$	Weights of the connections among the internal neurons
ESNs	Echo State Networks	$\boldsymbol{W}_{ofb}$	Weights of the connections from the output back to the reservoir internal neurons
FANNs	Feedforward Artificial Neural Networks	$\boldsymbol{W}_{out}$	Weights of the connections from the input and the reservoir internal neurons to the output
MODE	Multi-Objective Differential Evolution	$\boldsymbol{W}_{out}^{tf}$	Teacher forcing weights of the connections from the reservoir internal neurons to the output
SaNSDE	Self-adaptive Differential Evolution with Neighborhood Search	$\boldsymbol{x}(t)$	Vector of the activations of the reservoir neurons at generic time t
TOPSIS	Technique for Order of Preference by Similarity to Ideal Solution	$\boldsymbol{u}(t)$	*ESN* input vector

CRA	Cumulative Relative Accuracy	$y(t)$	*ESN* output vector
$\alpha-\lambda$	Alpha-Lambda metrics	a	Leaky rate
P	Number of measured signal	$f_{out}(\cdot)$	Activation function of the output neuron
R	Number of available run-to-failure trajectories	$\lvert\lambda_{max}\rvert$	Magnitude of the largest eigenvalue of $\boldsymbol{W}$
r	Index of a generic run-to-failure trajectory	ρ	Spectral radius
n^r	Number of events collected during the *r*-th run-to-failure trajectory	c	Connectivity
τ_j^r	Time of occurrence of the j-th event of the r-th run-to-failure trajectory	$\boldsymbol{IS}$	Scaling factor of $\boldsymbol{W}_{in}$
$\boldsymbol{z}^r(\tau_j^r)$	Vector of measurements collected at time τ_j^r from the *r*-th run-to-failure trajectory	OFB	Scaling factor of $\boldsymbol{W}_{ofb}$
$RUL^{test}(t)$	Ground-truth RUL of a test system at time *t*	L	System load
t_f^{test}	Ground-truth failure time of the test system	n_f	Number of failed components in the system
$\boldsymbol{z}^{test}(\tau_j)$	Vector of measurements collected from the test system at time τ_j	LS_t	Load sharing factor of the components at time t
n^{test}	Number of events collected during the life of the test system	d_t^i	*i*-th component degradation level at time t
$R\hat{U}L(t)$	RUL predicted by the model at time t	T_t	Temperature experienced by the components at time t
N_x	Number of reservoir internal neurons	T_t^{env}	Environment temperature at time t

1. Introduction

Prognostics aims at predicting the degradation of equipment for estimating the Remaining Useful Life (*RUL*) (Zio and Di Maio, 2010; Zio, 2012; Liao and Köttig, 2014; Palacios *et al.*, 2015; Prytz *et al.*, 2015; Lei *et al.*, 2018). Prognostic methods are classified into model-based and data-driven (Schwabacher and Goebel, 2007; Zio and Di Maio, 2010). Model-based approaches, which are based on the use of physics-based models of the degradation processes, are typically applied to safety-critical and slow-degrading equipment whose degradation mechanisms have been extensively studied (Cai, Huang and Xie, 2017, Peng *et al.*, 2018). By contrast, data-driven approaches are typically used when accurate physics-based models of the, possibly competing, degradation processes which the components of industrial systems are subjected to are not available (Hu, Youn and Kim, 2012; Medjaher, Tobon-Mejia and Zerhouni, 2012; Rigamonti *et al.*, 2017; Sardá-Espinosa, Subbiah and Bartz-Beielstein, 2017; Huang, Huang and Li, 2019). Since they require a large amount of run-to-failure degradation trajectories for model training (Hu *et al.*, 2012), data-driven approaches are typically applied to non-safety critical systems characterized by relatively short mean times to failure. They are distinguished into two approaches (*i*) degradation-based, which indirectly predict the system *RUL* by estimating the future evolution of the component degradation until a failure threshold is reached (Rigamonti, Baraldi, Zio, Astigarraga, *et al.*, 2016; Lim *et al.*, 2017) and (*ii*) direct *RUL* prediction-based, which predict the system *RUL* by developing a direct mapping from the condition monitoring signals to the system *RUL* (Khelif *et al.*, 2017). Although degradation-based approaches are closer to physics-based reasoning, they are more difficult to develop than direct *RUL* prediction-based approaches when the quantification of the component degradation and the identification of a failure threshold are not straightforward (Medjaher, Tobon-Mejia and Zerhouni, 2012).

Since prognostics requires to catch the dynamic behavior of the degradation process, static approaches based on the prediction of the equipment *RUL* at a given time as a function of the signal measurements at the same time typically provide unsatisfactory performances. An attempt to catch the system dynamics is to provide in input to the prognostic model the current and past signal values collected in a sliding time window. The main limitation of this approach is the difficulty in identifying a proper length of the time window, which allows representing the degradation dynamics without dramatically increasing the computational cost of the models, particularly if there are many input variables (Geraci and Gnabo, 2018). An alternative solution to the problem of learning the system dynamic for *RUL* prediction is the use of Recurrent Neural Networks (*RNNs*) (Zio, Broggi and Pedroni, 2009; Malhi, Yan and Gao, 2011; Guo *et al.*, 2017). The recurrent nature of *RNNs*, obtained by using feedback connections between the neurons of a layer and those of the preceding layers, allows processing dynamic information and makes them different from Feedforward Artificial Neural Networks (*FANNs*), which provide only a direct functional mapping between input and output data (Samanta and Al-Balushi, 2003; Moustapha and Selmic, 2008). Among the various types of recurrent networks that have been proposed in the last years, Echo State Networks (*ESNs*) are emerging due to their intrinsic dynamic properties, generalization capability, ability to handle noisy data and easiness of training (Jaeger, 2004). An extensive literature review on the use of *RNNs* in fault prognostics is reported in Section 2.

The objectives of the present work are (*i*) to predict the *RUL* of a system made by non-repairable interacting components, in which the measurements are collected only when triggering events, such as component faults or extreme operational conditions occur, and (*ii*) to estimate the uncertainty affecting the *RUL* prediction. These "snapshot" datasets are often encountered in industrial applications, dominated by the necessity of cost saving in storing and managing the databases (Weijters and van der Aalst, 2003; Liu, Li and Zio, 2017), and of reducing energy consumption and bandwidth sources (Tsividis, 2010). Since failure events rarely occur during the lifetime of a system, event-based datasets are dominated by the presence of a large number of missing measurements (Fink, Zio and Weidmann, 2015). Furthermore, the values of all the signals are missing at the same time. Given these characteristics, traditional methods for missing data management, e.g. case deletion (Schafer and Graham, 2002), imputation (Eekhout *et al.*, 2012; Ranjbar *et al.*, 2015; CH Cheng, CP Chan, 2019; Razavi-Far *et al.*, 2019) and maximum likelihood estimation (Baraldi and Enders, 2010), are difficult to apply. For instance, since Case Deletion methods discard patterns whose information is incomplete, they are not useful in case of event-based datasets where a pattern is either present or absent for all signals (Baraldi and Enders, 2010). Imputation techniques, which are based on the idea that a missing value can be replaced by a statistical indicator of the probability distribution generating the data, such as the signal mean (Donders *et al.*, 2006) or a value predicted by a multivariable regression model (Schafer and Graham, 2002), have been shown to be inaccurate in case of large fractions of missing values (Schafer, 1999; Vergouw *et al.*, 2010). Maximum Likelihood methods use the available data to identify the values of the probability distribution parameters with the largest probability of producing the sample data (Schafer, 1999). They typically require the Missing At Random (*MAR*) assumption, i.e. the probability of having a missing value is not dependent on the missing values (Little and Rubin, 2002; Donders *et al.*, 2006; Honaker and King, 2010), which is not met in event-based datasets.

To the best of our knowledge, few research works have considered fault prognostics in presence of missing data. A model based on Auto-Regressive Moving Average (*ARMA*) and an auto-associative neural networks, is developed for fault diagnostics and prognostics of water processes with incomplete data (Xiao *et al.*, 2017) and an hybrid architecture including physics-based and data-driven approaches are proposed to deal with missing data in case of rotating machinery (Leturiondo *et al.*, 2017). In the medical field, a Bayesian simulator is used to generate missing data for developing prognostic models (Marshall *et al.*, 2010) and a Multiple Imputation approach is used within a prognostic model for assessing overall survival of ovarian cancer in presence of missing covariate data (Clark and Altman, 2003). Notice that all these methods are based on the two successive steps of missing data reconstruction and prediction.

In this work, we consider the possibility of developing a method which is able to directly predict the equipment *RUL* without requiring a missing data reconstruction step. To this aim, the use of *ESNs* is considered due to their ability of maintaining information about the input history inside the reservoir state. The main difficulty to be addressed is that, contrarily to the typical applications of *ESNs*, the time intervals at which the data become available are irregular. Two different strategies are considered to cope with the event-based data collection. In *strategy 1*, the *ESN* receives an input pattern only when an event occurs. The pattern is formed by the measured signals and the time at which the event has occurred. In *strategy 2*, the reservoir states are excited at each time step. If an event has occurred, the reservoir states are excited both by the previous reservoir states and the measured signals, whereas, if an event has not occurred, they are excited only by the previous reservoir states. Therefore, it is expected that the connection loops in the reservoir allow reconstructing the dynamic degradation behavior of the system at those time steps in which events do not occur.

In both proposed strategies, a Multi-Objective Differential Evolution (*MODE*) algorithm based on a Self-adaptive Differential Evolution with Neighborhood Search (*SaNSDE*) (Yang, Tang and Yao, 2008) is used to optimize the *ESN* hyper-parameters. The Technique for Order of Preference by Similarity to Ideal Solution (*TOPSIS*) (Yoon and Hwang, 1995) is, then, used to select the optimal solution from the obtained *Pareto* solutions. Furthermore, a bootstrap aggregating (Bagging) ensemble method is applied to improve the *RUL* prediction accuracy and estimate the *RUL* prediction uncertainty. Given that *ESNs* cannot be fed by random sequences of patterns, the traditional Bagging sampling mechanism used to create the bootstrap training sets has been modified. In the proposed solution, the bootstrap training sets are obtained by concatenating entire run-to-failure trajectories, randomly sampled with replacement,

The two proposed strategies are applied to a synthetic case study, properly designed to mimic run-to-failure trajectories of a system of non-repairable interacting components in which the measurements are collected when events occur. The benefits of the proposed approaches are further shown by their application to a real-world case study concerning the prediction of the *RUL* of a sliding bearing of a turbine unit. The accuracy of the two strategies is evaluated considering the Cumulative Relative Accuracy (*CRA*) and Alpha-Lambda (*α-λ*) metrics

(Saxena *et al.*, 2010), and compared to that of a traditional feedforward neural network and a state-of-the-art ensemble of *ESN*s.

The original contributions of this work are:

1) the development of a novel method for fault prognostics in case of event-based data which does not require missing data reconstruction;
2) The adaptation of *ESNs* to deal with data collected at irregular time steps;
3) The modification of the sampling mechanism used by the Bagging method for creating the bootstrap training sets.

The remaining of this paper is organized as follows: Section 2 reports an extensive literature review on the use of *RNNs* in fault prognostics; Section 3 illustrates the work objectives and problem setting; Section 4 describes the proposed method for fault prognostics; Section 5 and 6 introduces the synthetic case study and the real-world case study and discusses the obtained results; finally, some conclusions and remarks are drawn in Section 7.

2 Recurrent Neural Networks in Fault Prognostics

Various types of *RNNs*, such as Long Short-Term Memory network (*LSTM*), Gated Recurrent Unit (*GRU*) *RNN*, have been used with success in prognostic applications (Zhang *et al.*, 2018; Chen *et al.*, 2019). A *RNN*-based model has been developed for predicting machine deterioration evolution using vibration data (Tse and Atherton, 1999). A Long Short-Term Memory (*LSTM*) *RNN* has been used to predict the remaining useful life of lithium-ion batteries (Zhang *et al.*, 2018). A *RNN* Encoder-Decoder network, which transforms multivariate time series subsequences into fixed-dimensional vectors, has been used to define health indicators and to predict the *RUL* of turbofan engines (Gugulothu *et al.*, 2017). A health indicator defined by using a *RNN* has been used for the prediction of bearing *RUL* (Guo *et al.*, 2017). Infinite impulse response locally recurrent neural networks has been employed for forecasting failures and predicting the reliability of components and systems (Zio *et al.*, 2012). Also, *RNNs* have been used in the domain of physical security information management (Rosenberg *et al.*, 2017; Tuor *et al.*, 2017; Azzouni and Pujolle, 2018; Rekik, Gransart and Berbineau, 2018; KP, 2019). *RNNs* have been used to build intrusion detection models for protecting from cyber-security threats (Kim *et al.*, 2016; Vinayakumar, Soman and Poornachandrany, 2017) and have been applied to cyber-security for decreasing malware attack losses (Tobiyama *et al.*, 2016; Rosenberg *et al.*, 2017; Teoh *et al.*, 2018; KP, 2019). A framework based on *LSTM RNNs* has been developed for traffic prediction in railway and used for information management and network security (De Bruin, Verbert and Babuska, 2017; Azzouni and Pujolle, 2018).

The main challenges for practical prognostic applications of *RNNs* are: *i)* the slow and computationally intensive training procedure, which cannot guarantee the final convergence of the algorithm towards an accurate and robust model (Jaeger *et al.*, 2007; Lukoševičius and Jaeger, 2009), and *ii)* the lack of guidelines for defining the *RNN* architecture, e.g. number of hidden layers and number of neurons.

To overcome these challenges, the Reservoir Computing (*RC*) paradigm has been proposed (Lukoševičius and Jaeger, 2009). *RC* involves randomly creating a *RNN*, called Reservoir, which remains unchanged during the training and is passively excited by the input signals. Among *RC* approaches, Echo State Networks (*ESNs*) have shown intrinsic dynamic properties, generalization capability and ability to handle noisy data (Jaeger, 2004). Considering *PHM* applications, *ESNs* have been mainly used for assessing component current health state and predicting health state evolution. An *ESN*-based echo state kernel recursive least squares algorithm is developed for tracking the health state of a turbofan engine (Zhou *et al.*, 2018). A genetic algorithm optimized double-reservoir echo state network has been developed for multi-regime time series prediction on turbofan engine (Zhong *et al.*, 2017). Echo State Networks have been used for the health monitoring of a test footbridge (Wootton, Day and Haycock, 2015). Fuel cell aging has been predicted using a method which combines *ESNs* and ANalysis Of VAriance (*ANOVA*) for the estimation of the importance of the model parameters (Morando *et al.*, 2015). With respect to *RUL* prediction, there are few applications of *ESNs*. An ensemble of *ESNs* has been developed for *RUL* prediction with uncertainty estimation, and applied to turbofan engines (Rigamonti, Baraldi, Zio, Roychoudhury, *et al.*, 2016). Also, *ESN*s have been used for predicting the *RUL* of industrial Fuel Cells (*FC*) (Morando *et al.*, 2017). It has been clarified that all these approaches focus on a single component and on signals measured at regular time steps. This has motivated the methodological development of the work.

3. Problem setting

This work assumes the availability of the measurements of *P* signals collected in correspondence of the occurrence of events during *R* run-to-failure degradation trajectories. The generic *r*-th trajectory is formed by the measurements collected at the occurrence of n^r events before the failure of the system. The time of occurrence of the *j*-th event of the *r*-th trajectory, $j = 1,\ldots,n^r$, will be referred to as τ_j^r and the corresponding measurement vector $\boldsymbol{z}^r(\tau_j^r)$. The objective of this work is to develop a direct data-driven prognostic method for the prediction of the ground-truth *RUL* of a test system at time *t*, $RUL^{test}(t) = t_f^{test} - t$, with t_f^{test} indicating the ground-truth failure time of the system, on the basis of the signal measurements $\mathbf{z}^{test}(\tau_j)$ collected in correspondence of the occurred events $j = 1,2,\ldots,n^{test}$, with $\tau_{n^{test}} < t$ indicating the last time at which signal measurements have been acquired. The *RUL* prediction at time *t* is indicated as $R\widehat{U}L(t)$.

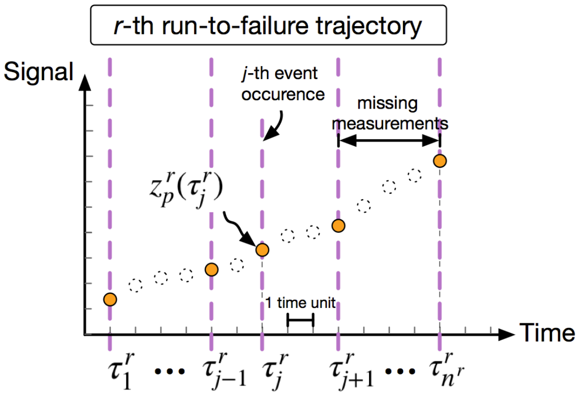
r-th run-to-failure trajectory
Signal
j-th event occurence
missing measurements
$z_p^r(\tau_j^r)$
1 time unit
Time
τ_1^r ••• τ_{j-1}^r τ_j^r τ_{j+1}^r ••• $\tau_{n^r}^r$

Figure 1. Example of data collected during a generic *r*-th run-to-failure trajectory. The vectors $\boldsymbol{z}^r = [z_1^r, \ldots, z_p^r, \ldots, z_P^r]^T$ contain the measurements collected at the times $\tau_1^r, \tau_2^r, \ldots, \tau_{n^r}^r$ of occurrence of events triggering the activation of the measurement system.

4. Methodology

The *ESN* is a *RNN* characterized by recurrent loops in its synaptic connection pathways (Jaeger, 2004). Differently from traditional *RNNs*, the internal neurons, which form the *ESN* reservoir, are sparsely connected (Lukoševičius and Jaeger, 2009). This architecture, which mimics the biological neural networks, allows maintaining an ongoing activation of the neurons even in absence of input to the *ESN* and, thus, it provides dynamic memory (Inubushi and Yoshimura, 2017).

The *ESN* architecture considered in this work is characterized by *P* input neurons, a reservoir with $N_x >> 1$ internal neurons and one output neuron representing the system *RUL* (Figure 2). Matrix $\boldsymbol{W}_{in}$ of size $N_x \times P$ contains the weights of the connections from the input neurons to the internal neurons, matrix $\boldsymbol{W}$ of size $N_x \times N_x$ contains the weights of the connections among the internal neurons, matrix $\boldsymbol{W}_{ofb}$ of size $N_x \times 1$ contains the weights of the connections from the output back to the reservoir internal neurons and matrix $\boldsymbol{W}_{out}$ of size $1 \times (P + N_x)$ contains the weights of the connections from the input and the reservoir internal neurons to the output. This work considers a reservoir with leaky-integrator neurons, whose state is updated according to:

$$\boldsymbol{x}(t) = (1-a)\boldsymbol{x}(t-1) + f(\boldsymbol{W}_{in}\,\boldsymbol{u}(t) + \boldsymbol{W}\boldsymbol{x}(t-1) + \boldsymbol{W}_{ofb}y(t-1)) \tag{1}$$

where $\boldsymbol{x}(t)$ is the activation vector of the reservoir neurons at the generic time *t*, $f(\cdot)$ is the internal neurons activation function, which is typically $tanh(\cdot)$, $\boldsymbol{u}(t)$ is the P dimensional input vector $\boldsymbol{u}(t) = \boldsymbol{z}^{test}(t)$, and $y(t) = R\widehat{U}L(t)$ is the output vector. The leaky rate $a \in [0,1]$ is a hyper-parameter controlling the decaying potential of neurons (Jaeger, 2005).

The output provided by the *ESN* at time *t* is:

$$y(t) = f_{out}(\boldsymbol{W}_{out} \cdot [\boldsymbol{u}(t)|\boldsymbol{x}(t)]) \tag{2}$$

where $f_{out}(\cdot)$ is the output neuron activation function, which is typically the identity function, $f(x) = x$ and the symbol $\cdot\,|\,\cdot$ represents the vertical concatenation operation.

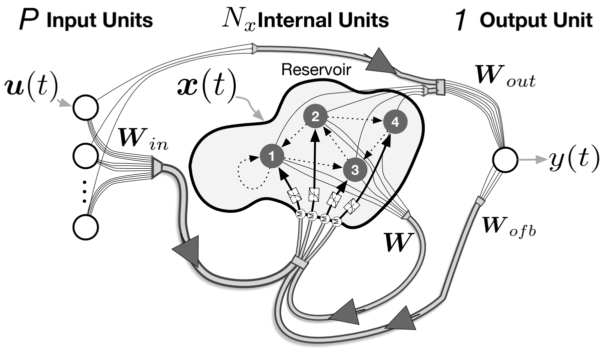

Figure 2. Echo State Network architecture: $\boldsymbol{u}(t)$ indicates the input vector, $\boldsymbol{x}(t)$ the reservoir state, $y(t)$ the output, at time t, $\boldsymbol{W}_{in}$ is the matrix of the weight connections from the P input units to the N_x internal units, $\boldsymbol{W}$ is the matrix of the weight connections among the N_x internal units, $\boldsymbol{W}_{out}$ is the matrix of the weight connections from the P input units and N_x internal units to the one-dimensional output unit, $\boldsymbol{W}_{ofb}$ is the matrix of the weight connections from the output unit to the N_x internal units.

The elements of $\boldsymbol{W}_{in}$, $\boldsymbol{W}$ and $\boldsymbol{W}_{ofb}$ are randomly sampled from uniform distributions according to the *RC* principles. To ensure the *Echo State Property* (Jaeger, 2004; Yildiz, Jaeger and Kiebel, 2012; Barancok and Farkas, 2014) which implies that the effect of the current states of the reservoir internal neurons and of the input on a future state vanishes gradually as time passes and does not amplify or persist, the reservoir connection matrix is typically scaled as $(\frac{\rho}{|\lambda_{max}|}) \cdot \boldsymbol{W}$, where $|\lambda_{max}|$ is the magnitude of the largest eigenvalue of $\boldsymbol{W}$ and $\rho \in (0,1)$ is a model hyper-parameter indicating the spectral radius. According to Jaeger (2010), the spectral radius is set to a value in the range $(0,1)$, for ensuring the *echo state property*.

The *ESN* training aims at finding optimal values for $\boldsymbol{W}_{out}$ and is performed by minimizing the quadratic error between the target output and the *ESN* output. To do that, the reservoir state $\boldsymbol{x}(t)$, the input $\boldsymbol{u}(t)$ and the output $y(t)$ are stacked into the matrixes $\boldsymbol{X}$, $\boldsymbol{U}$ and $\boldsymbol{Y}$, whose generic *t*-th column represents the reservoir state and the input and output acquired at time t. Then, $\boldsymbol{W}_{out}$ is obtained by solving the linear regression problem $[\boldsymbol{U}, \boldsymbol{X}] \rightarrow \boldsymbol{Y}$ (Lukoševičius and Jaeger, 2009).

4.1. ESN hyper-parameter setting

ESN main hyper-parameters are spectral radius ρ (Lukoševičius and Jaeger, 2009), leaky rate a (Jaeger *et al.*, 2007), connectivity c (Büsing, Schrauwen and Legenstein, 2010), reservoir size N_x, scaling factors of $\boldsymbol{W}_{in}$, $\boldsymbol{IS}$, and $\boldsymbol{W}_{ofb}$, OFB, (Jaeger *et al.*, 2007). The spectral radius determines how fast the influence of an input in a reservoir dies out with time (Jaeger *et al.*, 2007). The leaky rate controls the decaying potential of neurons: small values allow long retainment of past states (Jaeger *et al.*, 2007). Connectivity is defined by the ratio between the number of connections in the *ESN* reservoir and the number N_x^2 of all the possible connections. Small connectivity values characterize reservoirs with few connections among internal neurons, and, therefore, with a reduced capacity of representing the dynamic temporal behavior of the signals (Büsing, Schrauwen and Legenstein, 2010). On the other hand, the use of large connectivity values is limited by the associated large computational burdens. A proper setting of N_x is fundamental for obtaining accurate *ESN* predictions. According to Büsing, Schrauwen and Legenstein (2010), a critical *ESN* hyper-parameter is the reservoir size N_x: larger the reservoir size N_x, larger the reservoir Memory Capacity (*MC*) which quantifies the memory span of the *ESN*, i.e. its capability of encapsulating a certain input span within its internal states, being able to "remember" and recall it. If the connectivity satisfies $c \cdot N_x = M$ with $M \in \mathbb{R}^+$, i.e. the *ESN* connectivity is inversely proportional to the reservoir size, the *echo state property* is maintained and large reservoir $N_x >> 1$ can be used to guarantee proper memory capacity without extensively increasing the computational burden (Qiao *et al.*, 2016). The scaling factors $\boldsymbol{IS}$ depends on the degree of nonlinearity of the input-output relationship (Jaeger, 2010). If the neuron inputs are close to zero, the reservoir neurons, which have a $\tanh(\cdot)$ activation function, tend to provide outputs linearly dependent to the inputs, whereas when the neuron inputs are far from 0, the neurons tend to operate in a saturation zone of the activation function, where they exhibit more nonlinear behaviors with output values close to +1 or -1. The scaling of $\boldsymbol{W}_{ofb}$ is limited by an upper threshold at which the *ESN* starts exhibiting an unstable behavior, i.e. the output feedback loop starts to amplify the output creating a diverging generative mode (Jaeger, 2005).

Given the lack of guidelines for setting the *ESN* hyper-parameters and the importance of this task for obtaining stable and accurate *ESN* predictions, this work resorts to a Multi-Objective Differential Evolution (*MODE*)-

based approach. *MODE* is a parallel, direct, genetic algorithm-based search method which perturbs the current members of the population using the scaled differences of other two randomly selected members (Das and Suganthan, 2011). In particular, A Self-adaptive Differential Evolution with Neighborhood Search (*SaNSDE*) algorithm (Yang, Tang and Yao, 2008) is used, given its capabilities of automatically and adaptively setting the *MODE* control parameters, F and CR, and of escaping from local optima. Figure 3 shows the main steps of the *SaNSDE* search. A fixed random seed is set to initialize matrices $\boldsymbol{W}$, $\boldsymbol{W}_{in}$ and $\boldsymbol{W}_{ofb}$, whose values are kept fixed during *ESN* training. Thus, *ESN* hyper-parameters ρ, a, c, N_x, $\boldsymbol{IS}$, OFB univocally determine $\boldsymbol{W}$, $\boldsymbol{W}_{in}$ and $\boldsymbol{W}_{ofb}$. The initial population is obtained by randomly sampling the hyper-parameters values from uniform distributions on the ranges reported in Table 1. The spectral radius ρ is set in the range [0,1] to satisfy the *echo state property*. Leaky rate a is set in the range [0,1] according to the principle of reservoir computing (Jaeger *et al.*, 2007; Goudarzi, Shabani and Stefanovic, 2015; Inubushi and Yoshimura, 2017). The reservoir size N_x is set in the range [50,600], since larger reservoir sizes would lead to unfeasible computational burden (Barancok and Farkas, 2014). The input scaling factors $\boldsymbol{IS}$ and the Output Feedback (*OFB*) values are set in large ranges considering the typical values of the signals and the response of the activation function. Given the values of the signals and the use of a $tanh$ activation function, the ranges of the input scaling factors $\boldsymbol{IS}$ are taken equal to $[10^{-3}, 10^{3}]$, and the range of OFB is taken equal to $[10^{-5}, 10^{2}]$, being the output values larger than the input ones. More details of the *SaNSDE* algorithm can be found in (Yang, Tang and Yao, 2008).

The objectives of the search are the maximization of the accuracy metrics typically employed in prognostics, such as Cumulative Relative Accuracy (*CRA*) and Alpha-Lambda ($\alpha - \lambda$) (Saxena *et al.*, 2010). The *CRA* provides an estimation of the average relative *RUL* prediction error and, being a relative measure, tends to overestimate errors made at the end of the system life. The $\alpha - \lambda$ accuracy indicates how many times, on average, the *RUL* prediction falls within two relative confidence bounds. Notice that the computation of the *MODE* objective functions requires the division of the available run-to-failure trajectories into a training set used to optimize the *ESN* output weights $\boldsymbol{W}_{out}$ and a validation set used to estimate the CRA and $\alpha - \lambda$ metrics. Once the Pareto front solutions have been identified by the *SaNSDE* algorithm, the Technique for Order of Preference by Similarity to Ideal Solution (*TOPSIS*) is used to select among them a compromise solution, to be used for setting the *ESN* architecture (Behzadian *et al.*, 2012).

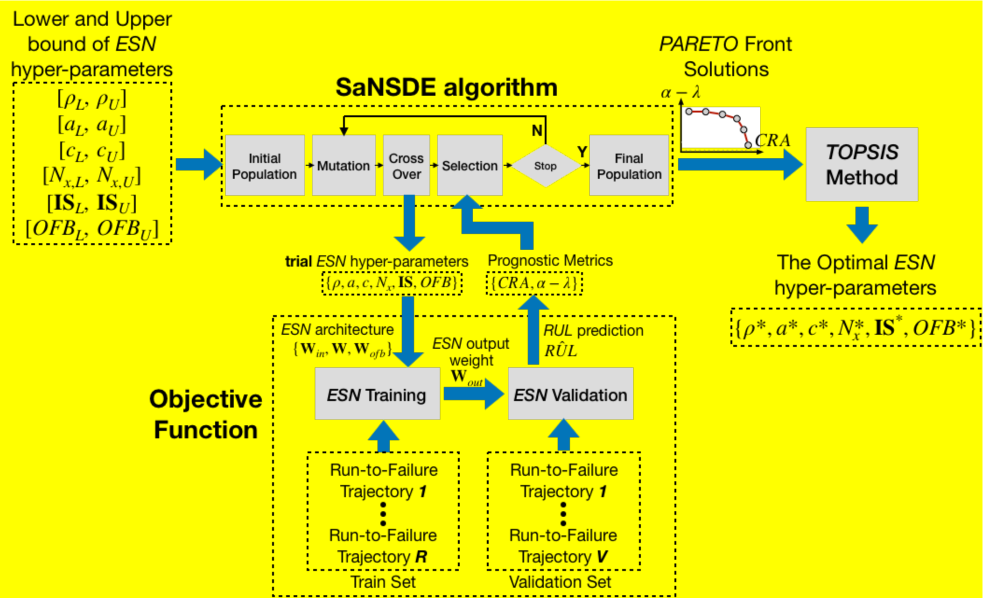
Lower and Upper bound of ESN hyper-parameters
$[\rho_L, \rho_U]$
$[a_L, a_U]$
$[c_L, c_U]$
$[N_{x,L}, N_{x,U}]$
$[\mathbf{IS}_L, \mathbf{IS}_U]$
$[OFB_L, OFB_U]$
SaNSDE algorithm
Initial Population
Mutation
Cross Over
Selection
Stop
N
Y
Final Population
PARETO Front Solutions
$\alpha-\lambda$
CRA
TOPSIS Method
The Optimal ESN hyper-parameters
$\{\rho^*, a^*, c^*, N_x^*, \mathbf{IS}^*, OFB^*\}$
trial ESN hyper-parameters
$\{\rho, a, c, N_x, \mathbf{IS}, OFB\}$
Prognostic Metrics
$\{CRA, \alpha-\lambda\}$
ESN architecture
$\{\mathbf{W}_{in}, \mathbf{W}, \mathbf{W}_{ofb}\}$
ESN output weight
$\mathbf{W}_{out}$
RUL prediction
$\widehat{RUL}$
Objective Function
ESN Training
ESN Validation
Run-to-Failure Trajectory 1
Run-to-Failure Trajectory R
Train Set
Run-to-Failure Trajectory 1
Run-to-Failure Trajectory V
Validation Set

Figure 3. Scheme of the overall procedure of optimization of the *ESN* hyper-parameters.

Table 1. Lower and upper values from which the parameters are sampled.

ESN hyper-parameters	[Lower Upper] values
Spectral radius (ρ)	$0 \sim 1$
Leaky rate (a)	$0 \sim 1$
Connectivity (c)	$(2 \sim 10)/N_x$
Reservoir Size (N_x)	$50 \sim 600$
Input Scaling factors (***IS***)	10^{-3}~10^{3}
Output feedback scaling (*OFB*)	10^{-5}~10^{2}

4.2. Strategies to deal with event-based measurements

Subsections 4.2.1 and 4.2.2 describe the two proposed strategies to deal with event-based measurements.

4.2.1. Strategy 1

The *ESN* is fed by a pattern only when an event occurs, i.e. at time $\tau_j, j = 1,2, \dots, n^r$. The $P+1$ dimensional input vector is made by the *P* signal measurements, $\boldsymbol{z}(\tau_j)$, and the time τ_j of the *j*-th event. The *ESN* output is the system *RUL* prediction, $R\widehat{U}L(\tau_j)$. The training set is obtained by concatenating the input-output data extracted from the *R* training run-to-failure trajectories, as shown in Figure 4. This strategy is similar to the one employed in the work of Choi *et al.*, (2015) who models event-based data characterized by temporal irregularity in the clinical event prediction research. Since no measurements are acquired at a generic time $t \in (t_j, t_{j+1})$, i.e. between the j-th and $j+1$-th events, the corresponding *RUL* prediction is:

$$R\widehat{U}L(t) = R\widehat{U}L(\tau_j) - (t - t_j) \qquad \forall\, t \in (t_j, t_{j+1}) \tag{3}$$

The *MODE* objectives considered for searching the optimal *ESN* hyper-parameter setting are mean *CRA* and mean $\alpha - \lambda$ computed over a set of validation trajectories.

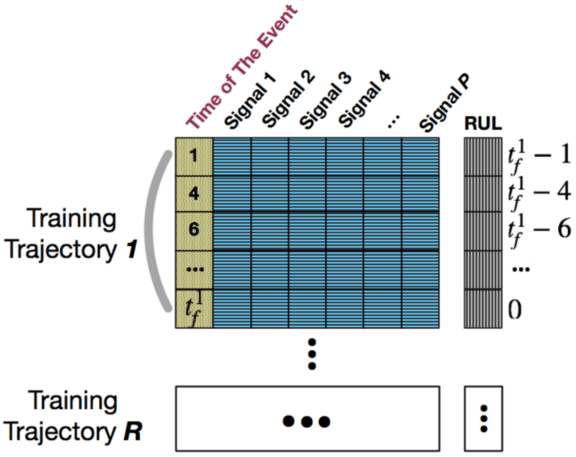

Figure 4. Input-Output training set used in *strategy 1*. t_f^1 indicates the ground-truth failure time of trajectory 1.

4.2.2. Strategy 2

The reservoir neurons are excited at each time step $t = 1,2, \dots, \tau_{n^{test}}$, independently from the occurrence of events in the monitored system. In particular, if at time t the event has not occurred and, therefore, the signal values $\boldsymbol{z}^r(t)$ are not measured, Equation 1 becomes (Figure 5):

$$\boldsymbol{x}(t) = (1-a)\boldsymbol{x}(t-1) + f(\boldsymbol{W}\boldsymbol{x}(t-1) + \boldsymbol{W}_{ofb}y(t-1)), \tag{4}$$

whereas, if at time t an event has occurred, i.e. $t = \tau_j^r, j = 1, \dots, n^r$, Equation 1 is applied. Notice that Equation 4 is used in the different context of *RNN* training by the teacher forcing method (Williams and Zipser, 1989; Lamb *et al.*, 2016) to speed up the convergence by forcing the reservoir to stay close to the ground-truth sequence. The pseudocode of the proposed algorithm is reported in the *Algorithm* (Figure 6). The training set is obtained by concatenating the input-output data according to the scheme shown in Figure 7. In this case, the *RUL* prediction, $R\widehat{U}L(t)$, is directly set equal to the *ESN* output $y(t)$. The considered *MODE* objectives are the mean *CRA* computed over a set of validation trajectories and the *CRA* measuring the prognostic accuracy of the *ESN* when it is used without input vectors:

$$R\widehat{U}L^{tf}(t) = f_{out}(\boldsymbol{W}_{out}^{tf} \cdot \boldsymbol{x}(t)) \tag{5}$$

where $\boldsymbol{W}_{out}^{tf} \in \mathbb{R}^{1 \times N_x}$ is obtained according to the principle of the teacher forcing method, by applying a linear regression of the target *RUL* sequence $RUL(t), t = 1,2, \dots, t_f^r$ and the corresponding reservoir state sequence $\boldsymbol{x}(t), t = 1,2, \dots, t_f^r$ obtained by Equation 4.

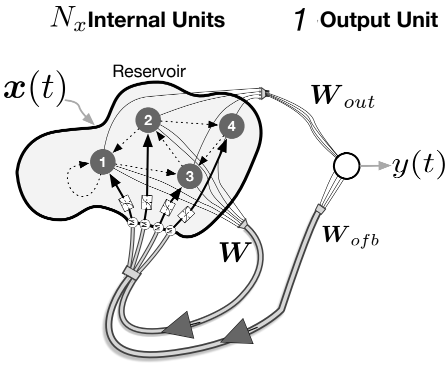

Figure 5. Reservoir operation in case of no event at time *t*.

```
Algorithm:
Input matrix $\boldsymbol{U}^r \in \mathbb{R}^{P\times t_f^r}$, Reservoir state matrix $\boldsymbol{X}^r \in \mathbb{R}^{N_x\times t_f^r}$, Output matrix $\boldsymbol{Y}^r \in \mathbb{R}^{1\times t_f^r}$.
Training Phase:
1:  for $r = 1,2,\dots,R$ do:
2:      for $j = 1,2,\dots,n^r$ do:
3:          $\boldsymbol{U}^r_{:,\tau_j^r} = \boldsymbol{z}^r(\tau_j^r)$
4:      end for
5:      for $t = 1,2,\dots,t_f^r$ do:
6:          if event occurs at time $t$ then:
7:              Use Equation 1 to update reservoir state $\boldsymbol{x}(t)$
8:          else
9:              Use Equation 4 to update reservoir state $\boldsymbol{x}(t)$
10:         end if
11:     end for
12:     Horizontally append $\boldsymbol{U}^r$ to $\boldsymbol{U}_{train}$, append $\boldsymbol{X}^r$ to $\boldsymbol{X}_{train}$ and append $\boldsymbol{Y}^r$ to $\boldsymbol{Y}_{train}$
13: end for
14: Do regression of $[\boldsymbol{U}_{train}|\boldsymbol{X}_{train}]$ onto $\boldsymbol{Y}_{train}$ to get $\boldsymbol{W}_{out}$
Testing Phase:
15: for $t = 1,2,\dots,t_f^{test}$ do:
16:     if event occurs at time $t$ then:
17:         use Equation 1-2 to obtain $R\widehat{U}L(t)$
18:     else
19:         use Equation 4 to obtain $\boldsymbol{x}(t)$
20:         use $y(t) = f_{out}(\boldsymbol{W}_{out}[\boldsymbol{0}|\boldsymbol{x}(t)])$ to obtain $R\widehat{U}L(t)$, where vector $\boldsymbol{0}$ has the same dimensionality of $\boldsymbol{u}(t)$.
21:     end if
22: end for
```

Figure 6. Pseudocode of *strategy 2*.

Lines 2-4 generate the input matrix $\boldsymbol{U}^r$ using the data in the generic *r*-th trajectory, whose τ_j^r-th column contains the measurement at event time τ_j^r. Lines 5-11 conditionally update the reservoir state $\boldsymbol{x}(t)$ by applying Equation 1 if an event occurs at time t, or Equation 4 otherwise. Once the reservoir state has been updated using all the data of the r-th run-to-failure trajectory, in Line 12, the input matrix $\boldsymbol{U}^r \in \mathbb{R}^{P\times t_f^r}$ is horizontally concatenated into the training matrix $\boldsymbol{U}_{train}$, the matrix $\boldsymbol{X}^r$ to $\boldsymbol{X}_{train}$ and the output matrix $\boldsymbol{Y}^r$ to $\boldsymbol{Y}_{train}$. Once all the training run-to-failure trajectories have been processed, the sizes of matrixes $\boldsymbol{U}_{train}$, $\boldsymbol{X}_{train}$, $\boldsymbol{Y}_{train}$ are $P\times(t_f^1+t_f^2+\cdots+t_f^R)$, $N_x\times(t_f^1+t_f^2+\cdots+t_f^R)$, $1\times(t_f^1+t_f^2+\cdots+t_f^R)$ respectively. Line 14 vertically concatenates $\boldsymbol{U}_{train}$ and $\boldsymbol{X}_{train}$ into the assembled matrix $[\boldsymbol{U}_{train}|\boldsymbol{X}_{train}]$ with size $(P+N_x)\times(t_f^1+t_f^2+\cdots+t_f^R)$ and find the matrix of weights $\boldsymbol{W}_{out}$ by solving the linear regression problem $[\boldsymbol{U}_{train}|\boldsymbol{X}_{train}]\rightarrow\boldsymbol{Y}_{train}$. Lines 16-22 report the algorithm used for estimating the *RUL* of a test trajectory. If an event occurs at time t, Equation 1 is used (Lines 16-18), otherwise Equation 4 is used (Lines 18-20).

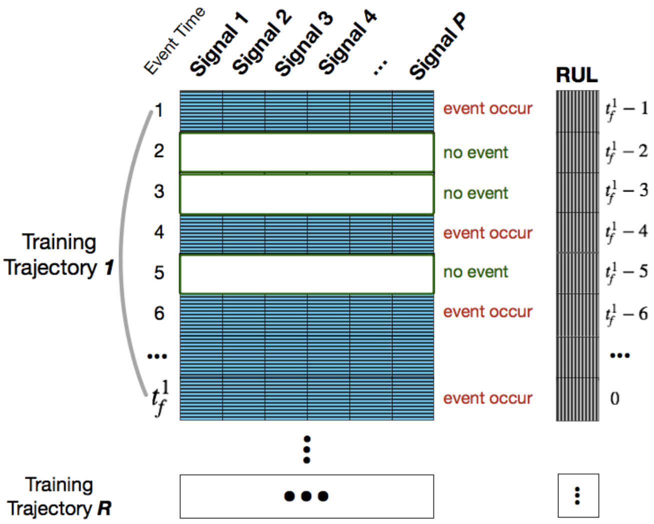

Figure 7. Input-Output training set used in *strategy 2*.

4.3. Ensemble of ESNs

An ensemble of *ESNs* is used in both *strategies 1* and *2* to improve model accuracy and estimate the uncertainty of the *RUL* predictions. Ensemble methods combine multiple models suited for a particular problem to generate an aggregated model, which can provide results more accurate than those provided by the individual models of the ensemble (Mendes-Moreira *et al.*, 2012). Bootstrap aggregating (Bagging) (Hauskrecht, 2004) and Boosting (Friedman, 2002) are two of the most commonly used approaches to build ensemble methods (Xing and Liu, 2019). This work considers Bagging, due to its capability of estimating the uncertainty of the prediction in the form of Prediction Intervals (*PI*) (Khosravi *et al.*, 2011a, 2011b). A prediction interval is an interval defined by a lower bound and an upper bound $[L_\alpha(t), U_\alpha(t)]$, in which the unknown value of the ground-truth *RUL* at time t, $RUL^{GT}(t)$, is expected to lie with a predetermined probability $(1-\alpha)$:

$$P\big(L_\alpha(t) < RUL^{GT}(t) < U_\alpha(t)\big) = 1-\alpha \tag{6}$$

Bagging is preferred to other methods for prediction interval estimation, such as Delta (Hwang and Ding, 1997; De Veaux *et al.*, 1998) and Bayesian Networks (Cai *et al.*, 2014, 2017; Cai, Huang and Xie, 2017), due to its simplicity, ease of implementation and reduced computational complexity (Khosravi *et al.*, 2011a, 2011b). Bagging is based on the bootstrap generation of B new training sets D_b, obtained by uniformly sampling with replacement the input-output patterns from the original training set D.

Since the use of recursive models does not allow providing in input to the models random sequences of patterns originated from different trajectories, the sampling mechanism has been modified. Each bootstrap training set $D_b, b=1,\dots,B,$ is obtained by randomly sampling entire run-to-failure trajectories, with replacement. The final *RUL* prediction of the ensemble is:

$$R\widehat{U}L^E(t) = \frac{1}{B}\sum_{b=1}^{B} R\widehat{U}L^b(t) \tag{7}$$

where $R\widehat{U}L^b(t)$ is the *RUL* predicted at time t by the *b*-th bootstrap model.

The Bagging approach reported in the Appendix has been used to estimate the *RUL* prediction interval.

To assess the performance of the Bagging approach in estimating effective prediction intervals, this work considers the *PI* coverage probability (*PICP*) and Normalized Mean PI Width (*NMPIW*) metrics. The former has a value in the range [0 1] and is the relative number of ground-truth *RUL* values lying in the *PI* $[L_\alpha(t), U_\alpha(t)]$ (Khosravi *et al.*, 2011a, 2011b):

$$PICP = 1/t_f \sum_{t=1}^{t_f} C(t) \text{ with } C(t) = \begin{cases} 1, RUL^{GT}(t) \in [L_\alpha(t), U_\alpha(t)] \\ 0, RUL^{GT}(t) \notin [L_\alpha(t), U_\alpha(t)] \end{cases} \tag{8}$$

NMPIW quantifies the average *PI* width normalized with respect to the target *RUL* (Khosravi *et al.*, 2011a, 2011b):

$$NMPIW = 1/t_f \sum_{t=1}^{t_f} \frac{U_\alpha(t) - L_\alpha(t)}{RUL^{GT}(t)} \tag{9}$$

A satisfied *PI* estimation should simultaneously have a large *PICP* value $[0,1]$ and a small *NMPIW* value $[0, +\infty]$.

5. Synthetic case study

5.1. Simulation step and data description

This case study, which is inspired from by the case study in (Al-Dahidi *et al.*, 2016), considers a system made by four non-repairable electrolytic capacitors working in parallel (Figure 8). The system load, L, is equally shared by the healthy components, i.e. the load on a healthy component is:

$$LS = L/(4 - n_f) \tag{10}$$

where $n_f \in \{0,1,2,3\}$ is the number of failed components. The system fails when the last operating component fails.

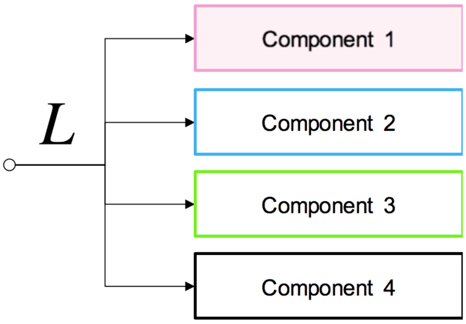

Figure 8. Structure of the system.

The degradation of the *i*-th component of the system is simulated by using (Rigamonti, Baraldi, Zio, Astigarraga, *et al.*, 2016):

$$d_t^i = d_{t-1}^i . e^{LS_{t-1} * C(T_{t-1})} + w_{t-1} \tag{11}$$

where d_t^i represents the *i*-th component degradation level at time t, LS_t the load of the *i*-th component at time t, w_t the process noise describing the degradation process stochasticity, and $C(T_t)$ is a function of the temperature experienced by the component at time t. Equation 11 is used by F. Perisse, *et al.*, (2006) to describe the degradation process of an electrolytic capacitor. Function $C(T_t)$ represents the degradation dependence from temperature, which is typically based on the Arrhenius law (Perisse *et al.*, 2006):

$$C(T_t) = \frac{ln\,(2)}{Life_{norm} \cdot \exp\left[\frac{E_a}{k} \frac{273 - T_t}{273 \cdot T_t}\right]} \tag{12}$$

where E_a is the activation energy of the component, k is the Boltzmann constant, and $Life_{norm}$ is the nominal life of the component aged at the constant nominal temperature 273 K.

The simulation of a system run-to-failure trajectory starts assuming at $t = 1$ a degradation level $d^i = 1$ (in arbitrary units) for all the system components, $i = 1, \dots ,4$, and proceeds by randomly sampling the noise w_{t-1} from a zero-mean Gaussian distribution with standard deviation σ_w and applying Equation 11. The temperature T_t experienced by the components at time t is influenced by the environment temperature T_t^{env} and the effect of the operational condition, which is represented by an additive temperature term Γ_t:

$$T_t = T_t^{env} + \Gamma_t \tag{13}$$

The environmental temperature is assumed to have a periodic triangular wave behavior, which reproduces its seasonality:

$$T_t^{env} = 2(T_{max}^{env} - T_{min}^{env})\left|\frac{t}{p} - floor\left(\frac{t}{p} + \frac{1}{2}\right)\right| + T_{min}^{env} \tag{14}$$

where T_{max}^{env} and T_{min}^{env} are the maximum and minimum annual environment temperatures, respectively, and p is the period of the environment temperature cycle. The term Γ_t is sampled from a uniform distribution in the range (1,9]. Figure 9 (top) shows a simulated evolution of the temperature experienced by the system during a run-to-failure trajectory.

A component fails when its degradation level d_t reaches the failure threshold, here set to 1.25, 1.50, 1.75 and 1.77 for components 1, 2, 3 and 4, respectively. The failure threshold values have been taken from (Venet, Darnand and Grellet, 1993; Al-Dahidi *et al.*, 2016; Rigamonti, Baraldi, Zio, Astigarraga, *et al.*, 2016), where values in the range [1.3 3] are reported for different types of electrolytic capacitors. Notice that the threshold values are independent on the operating mode Γ_t, which, on the other side, influences the evolution of the degradation process.

The events in correspondence of which the measurements are taken are:

1) when the temperature T_t^{env} experienced by the system exceeds either a lower or an upper temperature bound, set to $320K$ and $380K$ respectively;

2) when the degradation level of any one of the components exceeds their thresholds of 1.25, 1.50, 1.75, 1.77, respectively.

The following 6 measurements are assumed to be available in correspondence of events:

$$\mathbf{z}_i(\tau_j) = d_{m,\tau_j}^i = d_{\tau_j}^i \cdot \left(\alpha + \beta e^{-\frac{\left(T_{\tau_j} - 273\right)}{\gamma}}\right) + \eta_{\tau_j}, i = 1, \dots ,4, j = 1, \dots , n^r \tag{15}$$

$$\mathbf{z}_5(\tau_j) = T_{\tau_j},\ \mathbf{z}_6(\tau_j) = \Gamma_{\tau_j} \tag{16}$$

where η_{τ_j} represents the measurement noise which is sampled from a zero-mean Gaussian distribution with standard deviation σ_η, and α, β and γ are parameters characteristics of the component. Table 3 reports the values of the parameters in Equations 10-14 used for the system run-to-failure trajectory simulations, which are taken from (Rigamonti, Baraldi, Zio, Astigarraga, *et al.*, 2016). The activation energy E_a and parameters α, β, γ are set with reference to experimental laboratory tests on an electrolytic capacitor using a FLUKE

PM6306 RLC meter (Rigamonti, Baraldi, Zio, Astigarraga, *et al.*, 2016). The parameter $Life_{norm}$ is set equal to the expected life of the component at the nominal temperature, as in (Rigamonti, Baraldi, Zio, Astigarraga, *et al.*, 2016). The period of the environment temperature cycle p is set equal to one year (8760 hours), to simulate the seasonal temperature modifications.

Figure 9 shows the evolution of the measured quantities during one of the simulated system run-to-failure trajectories and the events at which measurements are collected (vertical lines).

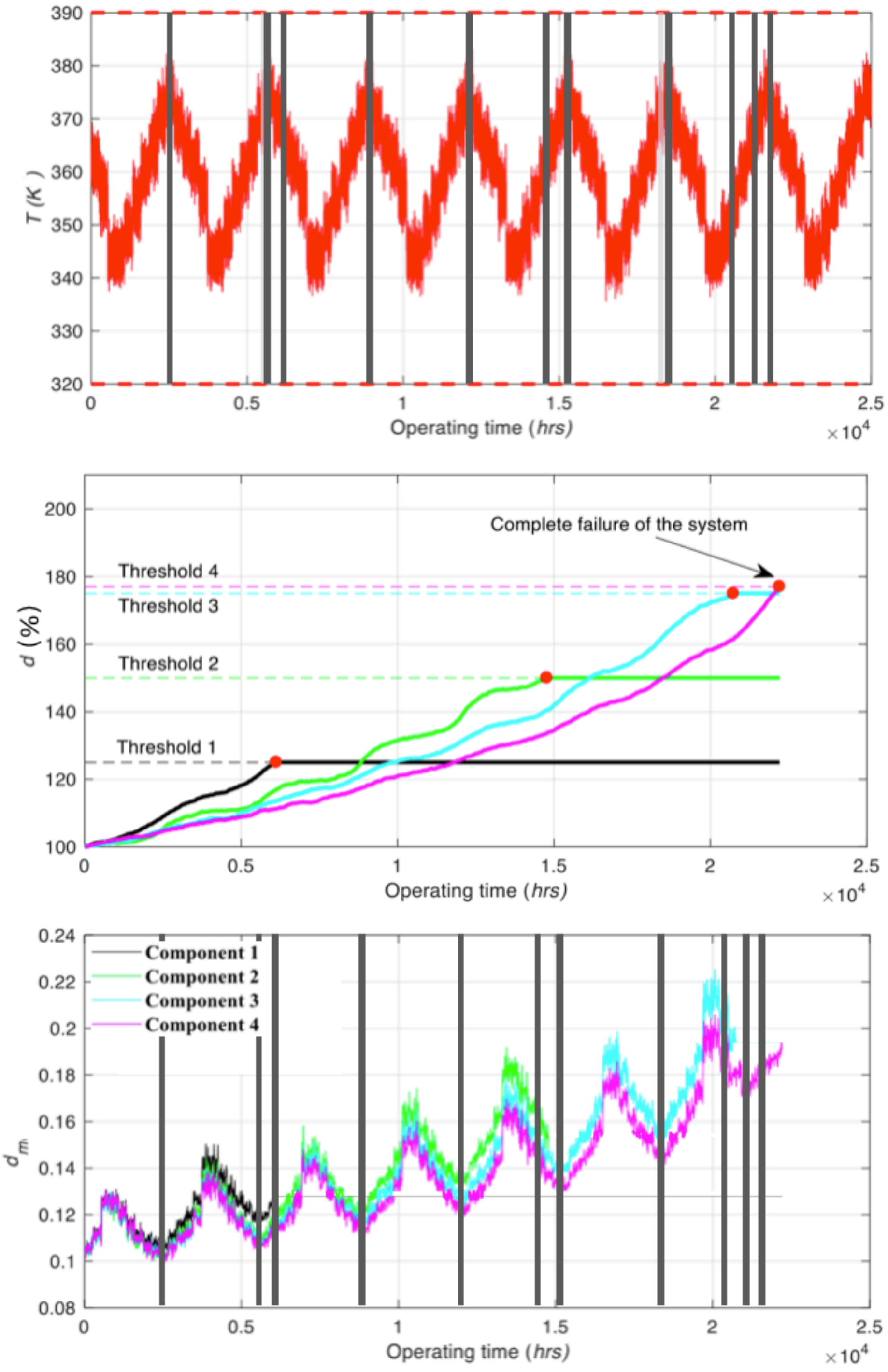

Figure 9. Simulation of a system run-to-failure trajectory: component degradation level d^i, i=1,…4 (center); degradation measurements d^i_{m,τ_j}(bottom); temperature (T_{τ_j}) (top), The vertical lines indicate the time at which the measurements are taken.

150 run-to-failure simulation trajectories, partitioned into a training (50 trajectories), a validation (50 trajectories) and a test (50 trajectories) sets have been generated. On average, an event generates the measurements of the 6 quantities every 55 time units. Thus, the dataset is characterized by a fraction of missing data equal to

nearly 98%. The simulated trajectories are characterized by large variability, with a minimum lifetime of 1.45×10^4 hours, a maximum lifetime of 3.5×10^4 hours and a obtained deviation of 1.0258×10^4 hours (Table 2).

Table 2. Statistical indicator of the system lifetimes.

Statistical indicator	Value [hours]
Average	2.36×10^4
Standard deviation	1.0258×10^4
Minimum	1.45×10^4
Maximum	3.5×10^4

Table 3. Parameters for the system run-to-failure trajectory simulations.

Parameters	Value
Standard deviation of w (σ_w)	0.002
Activation energy (E_a)	6.48×10^{-20}J
Boltzmann constant (k)	1.38×10^{-23}J/K
Component nominal life ($Life_{norm}$)	87600
Period of the environment temperature cycle (p)	8760
Standard deviation of η (σ_η)	0.002
Component characteristics (α)	0.0817
Component characteristics (β)	0.037
Component characteristics (γ)	30.682

5.2. Results and Discussion

The simulated measurements are linearly normalized in the range [0,1]. The hyper-parameters of the *ESNs* developed for *strategies 1* and *2* have been set by performing the *MODE* search described in Section 4.1 with a population of 500 chromosomes and considering 300 generations. The objective functions have been computed using the 50 run-to-failure trajectories of the validation set. A reservoir washing out procedure (Jaeger, 2005) is applied each time a new degradation trajectory is processed to avoid the influence of data collected from different systems on the neuron states (Lukoševičius and Jaeger, 2009).

Table 4. Optimal *ESN* hyper-parameters obtained for *strategies 1* and *2*.

ESN hyper-parameters	*Strategy 1 ESN*	*Strategy 2 ESN*
Spectral radius (ρ)	0.9276	0.4127
Leaky rate (a)	0.5073	0.5919
Connectivity (c)×Rerservoir Size (N_x)	7.5225	7.9843
Reservoir Size (N_x)	128	486
Input Scaling for Temperature (IS_T)	1.9030	6.4544
Input Scaling for Operation Mode (IS_Γ)	0.0543	18.5619
Input Scaling for Degradation level of component 1,2,3 and 4 (IS_D)	0.9070	13.3993
Output feedback scaling (*OFB*)	2.7970	1.3640

Table 4 reports the optimal hyper-parameters of *Strategy 1 ESN* and *Strategy 2 ESN*. Being the reservoir size of *strategy 2 ESN* (hereafter referred to a *ESN$_2$*) larger than *strategy 1 ESN* (hereafter referred to a *ESN$_1$*), *ESN$_2$* is characterized by a larger memory capacity (15 time units) than *ESN$_1$* (11 events). Since there is on average one event every 55 time units, the memory capacity of *ESN$_1$* is of about $11 \times 55 = 605$ time units which is much larger than that of *ESN$_2$*. All input scaling factors of *ESN$_2$* are larger than those of *ESN$_1$*. This is due to the fact that the input stream of *ESN$_1$* is continuous, which requires smaller input scaling factors to avoid internal neuron saturation. The spectral radius of ESN_2 is smaller than that of ESN_1, to decrease the risk of instability in presence of large input scaling factors. Figure 10 shows the *RUL* predictions obtained by ESN_1 and ESN_2 on two run-to-failure trajectories. According to Equation 3, the *RUL* predictions of ESN_1 linearly decreases with time in the time span between two consecutive events, τ_j and τ_{j+1}, whereas ESN_2 provides a *RUL* prediction at each time step by using the *Algorithm* proposed in Figure 6 (Lines 16-21). Notice that despite the variability of the trajectory lengths (Table 2), the two strategies are able to accurately predict the *RUL*.

Table 5 compares the accuracy metrics CRA and $\alpha - \lambda$ of the developed ESN_1, ESN_2, *strategy 1 Ensemble* (hereafter referred to as Ens_1), and *strategy 2 Ensemble* (hereafter referred to as Ens_2), with that of a Feedforward Artificial Neural Network (*FANN*) and of an ensemble of traditional *ESNs* (hereafter referred to as *Ens*). The *FANN* architecture, which is characterized by two hidden layers with ten neurons each, has been set by trial and error using the random search algorithm (Bergstra, James and Bengio, Yoshua, 2012). The hyper-parameters of the *ESNs* of the literature approach have been set using the procedure described in Section 4.1. The ensembles *Ens* Ens_1 and Ens_2 have been created by generating $B = 50$ *ESN* bootstrap models. The CRA and $\alpha - \lambda$ metrics have been computed on the 50 run-to-failure trajectories of the test set. The *FANN* predicts the *RUL* at time τ_j, using the signal measurements $\boldsymbol{z}(\tau_j)$.

Notice that ESN_1, ESN_2, Ens_1 and Ens_2 provide more accurate predictions than *FANN* and *Ens*. Figure 10 shows that the *RUL* predictions of *FANN* are less satisfactory at the end of the system life, when the use of the historical degradation information becomes more relevant. Ens_1 provides more accurate and stable *RUL* predictions than Ens_2, given the different performances of the ensemble individual models. Ens_1 and Ens_2 tend to provide more accurate predictions than *Ens* at the end of the run-to-failure trajectories, when the presence of event-based measurements makes the latter ensemble unstable. Furthermore, *strategy 1* outperforms *strategy 2* in all cases, i.e. when it is applied to both a single *ESN* and to an ensemble of *ESNs*. This is due to the fact that the reservoir of ESN_2 tends to gradually forget during the time span between two consecutive events, τ_j and τ_{j+1}, when no measurements are acquired. In practice, since ESN_2 reservoir has a *MC* equal to 15 time units and the measurements are acquired on average every 55 time units, the *ESN* is not able to retain the collected information for enough time. The reservoir forgetting phenomenon is shown in Figure 11: the occurrence of an event modifies the neuron state for a short interval of time after which they return to a stable value.

A possible solution to increase the memory capacity of ESN_2 *reservoir* is to further increase the number of internal neurons, N_x. Since the reservoir memory capacity, *MC*, tends to increase with the logarithm of the number of internal neurons, N_x, (Büsing, Schrauwen and Legenstein, 2010), the use of large reservoirs is in practice unfeasible for the computational efforts necessary for their construction.

Table 6 reports the metrics *PICP* and *NMPIW* used to assess the performance of the proposed methods for the identification of prediction intervals of Ens_1 *and* Ens_2. Figure 12 shows the 95% *PI* of Ens_1 and Ens_2. Notice that the coverage of both methods is satisfactory, being larger than the target value 0.95, although ESN_2 tends to provide better coverage probability, ESN_1 has narrower *PI*.

Table 5. Comparison of the performances of the proposed approaches (*ESN₁*, *ESN₂*, *Ens₁*, *Ens₂*) and the state-of-the-art approaches (*FANN*, *Ens*). *CRA* ranges from -∞ to 1, large *CRA* values indicating superior performances. $\alpha-\lambda$ ranges from 0 to 1, large $\alpha-\lambda$ values indicating superior performance.

RUL prognostic metric	CRA	$\alpha-\lambda$
Proposed approaches:		
ESN₁	0.7114±0.1408	0.6350±0.2765
ESN₂	0.4920±0.2094	0.4982±0.1696
Ens₁	**0.7156±0.1395**	**0.6362**±0.2715
Ens₂	0.4997±0.2101	0.4997±**0.1680**
State-of-the-art approaches:		
FANN	0.0191±0.2255	0.4634±0.1995
Ens	0.4261±0.3340	0.5963±0.1544

Table 6. Prediction intervals metrics of the two proposed *ESN strategies*.

Prediction interval 95% confidence	*PICP*	*NMPIW*
Ens₁	0.95	**1.1222**
Ens₂	**0.97**	1.6838

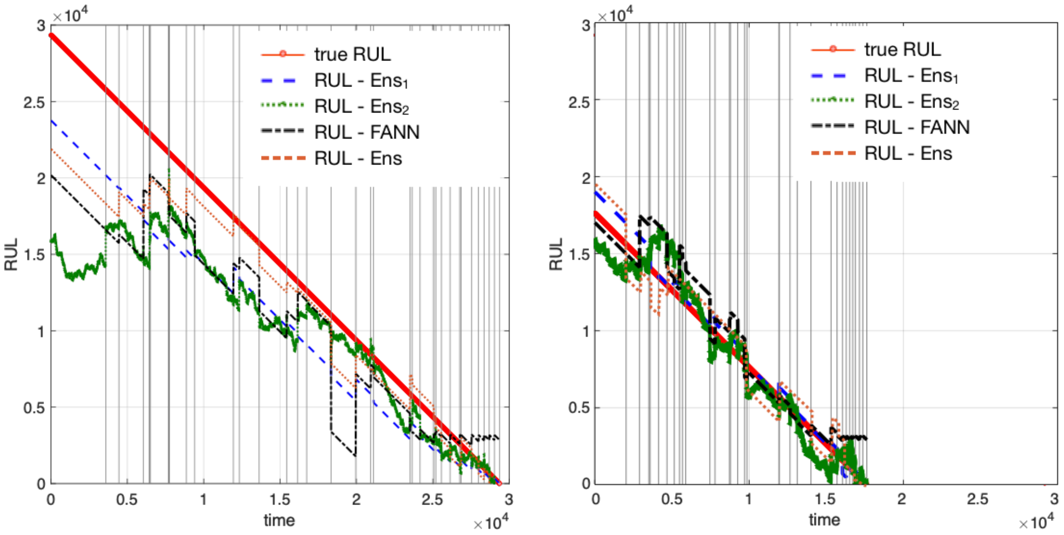

Figure 10. Remaining Useful Life (*RUL*) predictions on two run-to-failure test trajectories. The vertical lines indicate the time at which the measurements are taken.

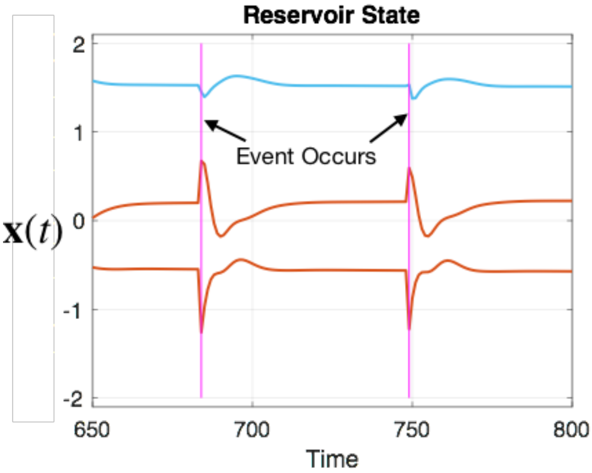

Figure 11. Time evolution of the internal states of three reservoir neurons of *ESN₂*.

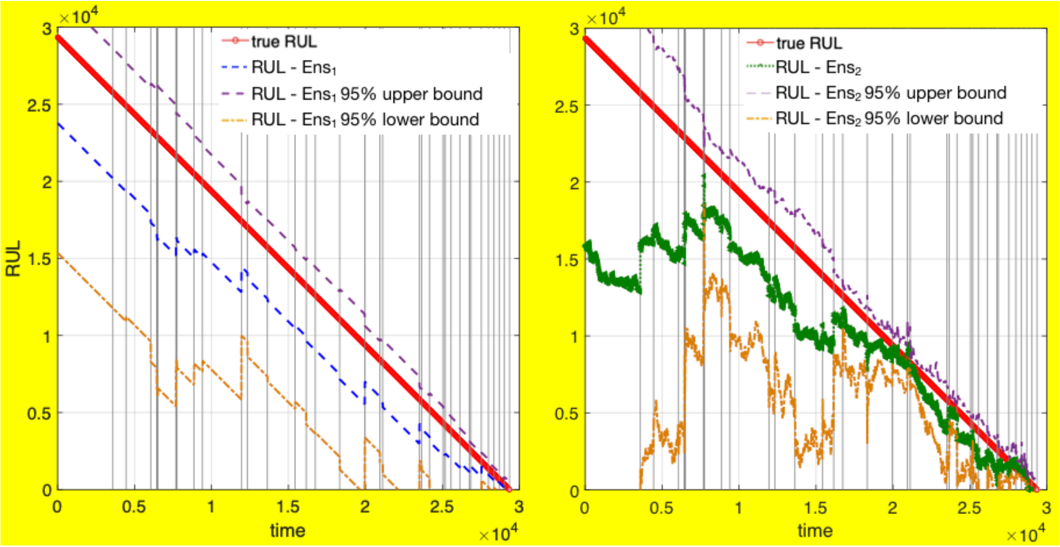

Figure 12. Remaining Useful Life (*RUL*) predictions intervals of. Ens_1 (left) and Ens_2 (right) on a run-to-failure test trajectory. The vertical lines indicate the time at which the measurements are taken.

6. Case study II: real industrial case

6.1. Data Description

This case study considers the degradation and failure process of bearings used in large turbine units. The dataset, which can be download from (Dataset 2019), contains signal measurements collected from five turbine units, U1, U2, U3, U4 and U5. Vibration signals are measured by a set of seven sensors. which include four eddy current displacement sensors measuring the radial vibration of the rotor at both ends, two eddy current displacement sensors measuring axial vibration of the rotor and one sensor measuring the unit rotating speed. Measurements are triggered by abnormal behaviors of the units, such as large environmental noise and anomalous vibration behavior. A number between 200 and 300 waves is collected for a time interval of around 10 minutes by each sensor with a sampling frequency of 3.2 kHz. Each wave is composed by 1024 samples. The acquisition time of all sensors is synchronized. Figure 13 shows an example of signal measurements performed during a run-to-failure trajectory. Notice that the anomalies occur at irregular time period, whereas after each anomaly measurements package are acquired at regular frequency.

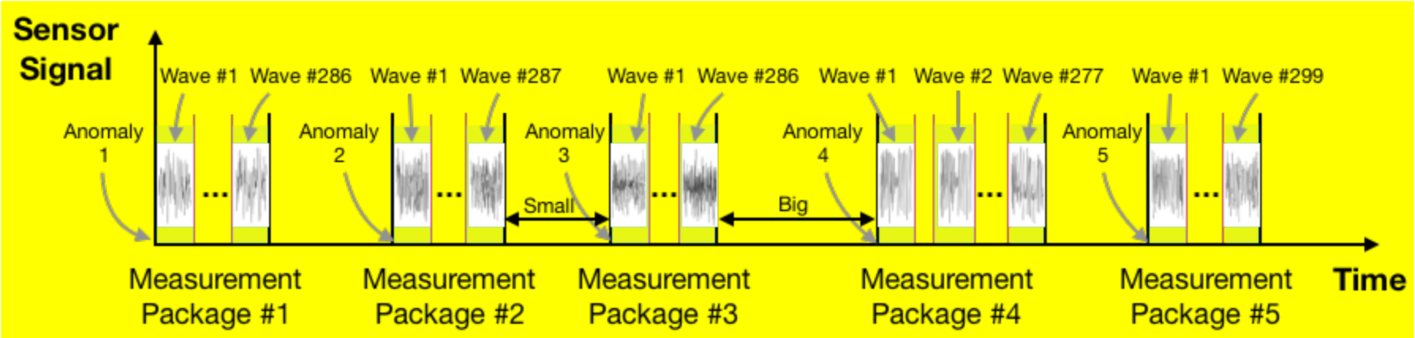

Figure 13. Measurement collection scheme.

6.2. Data preprocessing

To reduce the dimension of the 1024 values collected in the time domain during a single wave and the effects of phase shifts in different acquisitions, two features, i.e. the energy and the Shannon entropy, are extracted from each wavelet to characterize the bearing degradation, according to Heidari Bafroui and Ohadi (2014).

6.3. Results and Discussion

The hyper-parameters of *ESN₁* and *ESN₂* have been set by performing the *SaNSDE* search described in Figure 3 (Section 4.1) with a population of 100 chromosomes, 50 generations and the hyper-parameter ranges reported in Table 1. The objective function has been computed using a leave-one-out cross validation procedure. The reservoir washing out procedure described in Section 5.2 is applied each time a new degradation trajectory is processed. Table 7 reports the obtained optimal *ESN* hyper-parameters. Similar to the synthetic case study of Section 5, *ESN₁* is characterized by smaller reservoir size than that of ESN_2. Differently, the leaky rate of *ESN₂* is smaller than that of *ESN₁* and the spectral radius of *ESN₂* larger than that of *ESN₁*. These differences among the two case studies are due to the very different average number of consecutive missing values, which is nearly 55 in the synthetic case study and nearly 1000 in the present case study. The very large number of consecutive missing measurements causes an increase of the memory capacity and a decrease of the leaky rate.

Table 7. Optimal *ESN* hyper-parameters of *ESN₁* and *ESN₂* for real industrial case study.

ESN hyper-parameters	*ESN₁*	*ESN₂*
Spectral radius (ρ)	0.0866	0.9999
Leaky rate (a)	0.7825	0.072
Connectivity (c)×Rerservoir Size (N_x)	9.9044	6.8276
Reservoir Size (N_x)	156	228
Input Scaling for radial & vibration sensors signal (**IS_V**)	9.2652	1.7027
Input Scaling for speed (IS_S)	8.4118	1.4733
Output feedback scaling (*OFB*)	7.8798	3.3929

Table 8 compares the accuracy metrics CRA and $\alpha - \lambda$ of *ESN₁*, *ESN₂, Ens₁*, *Ens₂*, *FANN* an *Ens*. The metrics are evaluated resorting to a twice nested leave-one-out cross validation approach, in which the outer loop has been used to compute the metrics and the inner loop to optimize the *ESN* hyper-parameters. Figure 14 shows the obtained *RUL* predictions. Table 8 shows that *ESN₁*, *ESN₂*, *Ens₁*, *Ens₂* provide a more accurate *RUL* prediction than *FANN*. The *Ens* overperforms *ESN₂* and *Ens₂*, given the large memory required by *strategy 2* to learn the *RUL* evolution pattern during the period of missing measurements. Given the very large average number of consecutive missing values, a very large reservoir size is needed, which is, in practice, unfeasible due to the computational efforts necessary for its construction.

Table 9 reports the metrics *PICP* and *NMPIW* of *Ens₁* and *Ens₂*. *Ens₂* shows a more satisfactory *PICP* but a less satisfactory *NMPIW* than *Ens₁.* Figure 15 shows the 95% *PI* confidence intervals. It is worth noting that the *PI* of *Ens₂* become narrower as time passes because the teacher forcing method of *ESN₂* can learn the evolution of *RUL* when the measurement is not available.

Table 8. Comparison of the performances of the proposed approaches (*ESN₁*, *ESN₂*, *Ens₁*, *Ens₂*) and the state-of-the-art approaches (*FANN*, *Ens*).

RUL prognostic metric	CRA	$\alpha - \lambda$
Proposed approaches:		
ESN₁	0.2452±0.4773	0.6446±0.1906
ESN₂	0.0523±0.5650	0.5173±0.2816
Ens₁	**0.2796**±0.5416	**0.6547**±0.2114
Ens₂	0.0998±**0.2720**	0.4718±**0.1151**
State-of-the-art approaches:		

FANN	-0.3155±0.6762	0.2001±0.2462
Ens	0.1462±0.5289	0.6417±0.2085

Table 9. Prediction intervals metrics of the two proposed *ESN strategies*.

Prediction interval 95% confidence	*PICP*	*NMPIW*
Ens$_1$	0.8606	**2.6940**
Ens$_2$	**0.8655**	2.9715

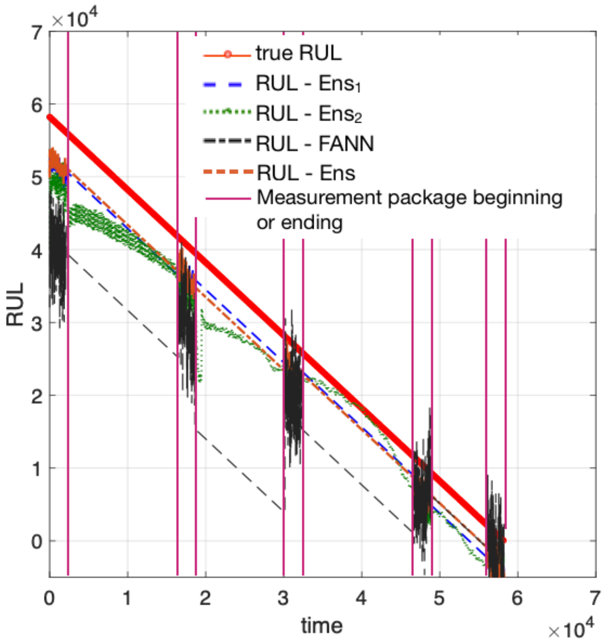

Figure 14. Remaining Useful Life (*RUL*) predictions on a run-to-failure test trajectory. The vertical lines indicate the time at which the measurements collection begins or ends.

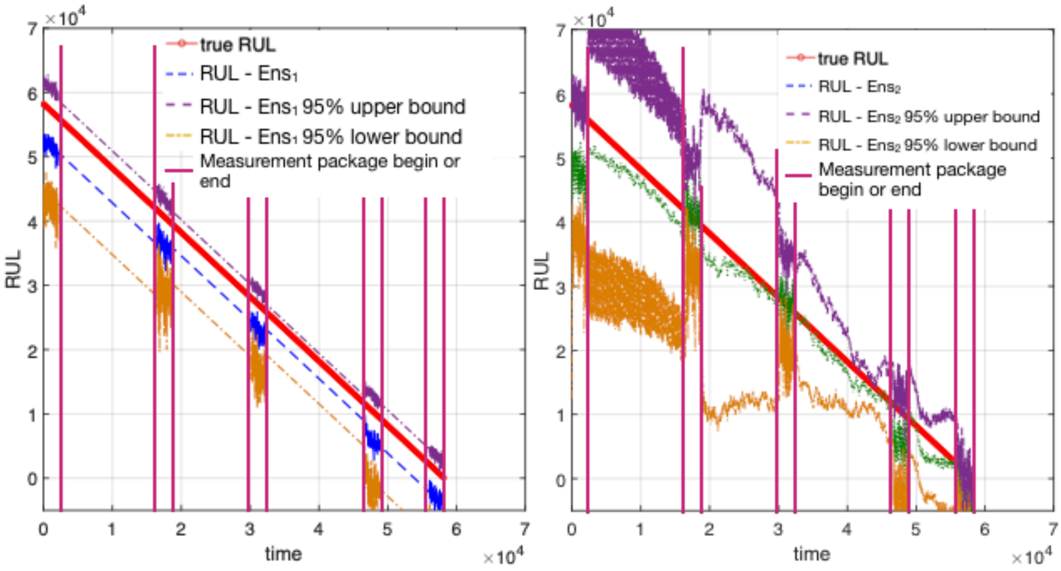

Figure 15. Remaining Useful Life (*RUL*) predictions intervals of. *Ens*$_1$ (left) and *Ens*$_2$ (right) on a run-to-failure test trajectory. The vertical lines indicate the time at which the measurements are taken.

7. Conclusion

Two strategies for predicting the remaining useful life of a system of non-repairable interacting components in case of data collected at irregular time step have been proposed. They are based on the use of a Bagging ensemble of *ESNs*, which have been adapted to deal with data collected at irregular time steps. In *Strategy 1,* the *ESNs* use the time of the measurement as additional input, whereas in *Strategy 2* the *ESN* reservoir neurons internal states at the time in which the measurements are not collected are excited only by the previous time internal states. A Self-adaptive Differential Evolution with Neighborhood Search (*SaNSDE*) algorithm is employed to set the *ESN* hyper-parameters. The two *ESN strategies* have been verified using a synthetic case study which mimics the behavior of a system in which measurements are collected only when events at the system components or extreme operational conditions occur and a real-world case study concerning the prediction of the *RUL* of a sliding bearing of a turbine unit. The obtained results have shown that both the proposed *ESN strategies* overperform a *FANN* and a state-of-the-art ensemble of *ESNs* and that *strategy 1* provides more satisfactory performances than *strategy 2*. The main drawback of *strategy 2* is that the memory capacity of the obtained *ESN* is not enough large and, therefore, the reservoir tends to forget the system state in the time span between two events.

Acknowledgement

This research is funded by INAIL - Grant 3158/2017 – 3rd Call SAF€RA. Mingjing Xu gratefully acknowledges the financial support from the China Scholarship Council (No. 201606420061).

Appendix. *RUL uncertainty estimation by Ensemble of ESNs*

With respect to the estimation of the *RUL* prediction interval, Bagging assumes that the prediction error, $\epsilon_t = RUL^{GT}(t) - R\hat{U}L^{E}(t)$, is a random variable with zero-mean distribution, whose variance $\sigma^2_{\hat{\epsilon}_t}$ need to be estimated. As in Baraldi, Mangili and Zio, (2013), $\sigma^2_{\hat{\epsilon}_t}$ can be decomposed into three terms: $\sigma^2_{d_t}$, the variance related to the uncertainty in the degradation process, $\sigma^2_{m_t}$, the variance related to the measurement noise and $\sigma^2_{R\hat{U}L^{E}(t)}$, the variance related to the imprecision of the model:

$$\begin{aligned}\sigma^2_{\hat{\epsilon}_t} &= E\left[\left(RUL^{GT}(t) - R\hat{U}L^{E}(t)\right)^2\right] \\ &= E\left[\left(RUL^{GT}(t) - \mu_{RUL(t)|d_t}\right)^2\right] + E\left[\left(\mu_{RUL(t)|d_t} - \mu_{RUL(t)|\boldsymbol{z}(t)}\right)^2\right] + E\left[\left(\mu_{RUL(t)|\boldsymbol{z}(t)} - R\hat{U}L^{E}(t)\right)^2\right] \\ &= \sigma^2_{d_t} + \sigma^2_{m_t} + \sigma^2_{R\hat{U}L^{E}(t)} \qquad \text{(A.1)}\end{aligned}$$

where the quantity $\mu_{RUL(t)|d_t}$ is introduced to represent the expected value of the *RUL* of equipment with degradation *d* at time *t* and $\mu_{RUL(t)|\boldsymbol{z}(t)}$ represents the expected value of the *RUL* of equipment from which the observation $\boldsymbol{z}(t)$ at time t has been obtained. Notice that it is assumed that the different components of the prediction error are independent. Then, the variance $\sigma^2_{R\hat{U}L^{E}(t)}$ can be estimated by (Khosravi *et al.* 2011a):

$$\sigma^2_{R\hat{U}L^{E}(t)} = \frac{1}{B-1}\sum_{b=1}^{B}\left(R\hat{U}L^{b}(t) - R\hat{U}L^{E}(t)\right)^2 \qquad \text{(A.2)}$$

and the variance $\sigma^2_d + \sigma^2_m$ can be approximated by:

$$\sigma_d^2 + \sigma_m^2 \approx E\left[\left(RUL^{GT} - R\widehat{U}L^E\right)^2\right] - \sigma_{R\widehat{U}L^E}^2 \tag{A.3}$$

Since RUL^{GT} is not directly measurable, in order to estimate the unknow value of $\sigma_d^2 + \sigma_m^2$, a set of variance-squared residuals is constituted by (Khosravi *et al.* 2011a):

$$\theta_t^2 = max\left(\left(RUL^{GT}(t) - R\widehat{U}L^E(t)\right)^2 - \sigma_{R\widehat{U}L^E(t)}^2, 0\right) \tag{A.4}$$

to form a new dataset $\{(\boldsymbol{z}^r(t), (\theta_t^r)^2)\}_{t=1,\ldots,t_f^r, r=1,\ldots,R}$, which is used to train a dedicated Feedforward *ANN*. The cost function for training the dedicated *ANN* is (Khosravi *et al.* 2011a):

$$cost = \frac{1}{2}\sum_{t=1}^{t_f}\left[\ln\left(\sigma_{\hat{\mu}_t}^2\right) + \frac{\theta_t^2}{\sigma_{\hat{\mu}_t}^2}\right] \tag{A.5}$$

where θ_t^2 is the target variance-squared residual for the *ANN* training and $\sigma_{\hat{\mu}_t}^2 = \sigma_{d_t}^2 + \sigma_{m_t}^2$ is the variance estimated by *ANN*. Note that the *ANN* output node activation function is selected to be an exponential function in order to enforce a positive value of $\sigma_{\hat{\mu}_t}^2$ and the *ANN* input is the reservoir state. The minimization of the cost function can be done using stochastic gradient descent methods.

Finally, the *PI* with confidence interval equal to $(1-\alpha)$ is (Baraldi, Mangili and Zio, 2013):

$$[L_\alpha(t), U_\alpha(t)] = [R\widehat{U}L^E(t) - z_\alpha\sigma_{\epsilon_t},\ R\widehat{U}L^E(t) + z_\alpha\sigma_{\epsilon_t}] \tag{A.6}$$

where z_α is the parameter representing the $\alpha/2$ quantile of a Student's t-distribution with number of degrees of freedom equal to the number B of bootstrap models.

Reference

Al-Dahidi, S. *et al.* (2016) 'Remaining useful life estimation in heterogeneous fleets working under variable operating conditions', *Reliability Engineering and System Safety*, 156, pp. 109–124. doi: 10.1016/j.ress.2016.07.019.

Azzouni, A. and Pujolle, G. (2018) 'NeuTM: A neural network-based framework for traffic matrix prediction in SDN', in *IEEE/IFIP Network Operations and Management Symposium: Cognitive Management in a Cyber World, NOMS 2018*. doi: 10.1109/NOMS.2018.8406199.

Baraldi, A. N. and Enders, C. K. (2010) 'An introduction to modern missing data analyses', *Journal of School Psychology*. doi: 10.1016/j.jsp.2009.10.001.

Baraldi, P., Mangili, F. and Zio, E. (2013) 'Investigation of uncertainty treatment capability of model-based and data-driven prognostic methods using simulated data', *Reliability Engineering and System Safety*. doi: 10.1016/j.ress.2012.12.004.

Barancok, P. and Farkas, I. (2014) 'Memory capacity of input-driven echo state networks at the edge of chaos', *Proceedings of the Interna- tional Conference on Artificial Neural Networks (ICANN)*, (Mc), pp. 41–48. doi: 10.1007/978-3-319-11179-7_6.

Behzadian, M. *et al.* (2012) 'A state-of the-art survey of TOPSIS applications', *Expert Systems with Applications*. doi: 10.1016/j.eswa.2012.05.056.

Bergstra, J. and Bengio, Y. (2012) 'Random search for hyper-parameter optimization', *Journal of Machine Learning Research*.

Dataset (2019). http://www.industrial-bigdata.com/datasets

De Bruin, T., Verbert, K. and Babuska, R. (2017) 'Railway Track Circuit Fault Diagnosis Using Recurrent Neural Networks', *IEEE Transactions on Neural Networks and Learning Systems*. doi: 10.1109/TNNLS.2016.2551940.

Büsing, L., Schrauwen, B. and Legenstein, R. (2010) 'Connectivity, Dynamics, and Memory in Reservoir Computing with Binary and Analog Neurons', *Neural Computation*, 22(5), pp. 1272–1311. doi: 10.1162/neco.2009.01-09-947.

Cai, B. *et al.* (2014) 'Multi-source information fusion based fault diagnosis of ground-source heat pump using Bayesian network', *Applied Energy*. doi: 10.1016/j.apenergy.2013.09.043.

Cai, B. *et al.* (2017) 'A Data-Driven Fault Diagnosis Methodology in Three-Phase Inverters for PMSM Drive Systems', *IEEE Transactions on Power Electronics*. doi: 10.1109/TPEL.2016.2608842.

Cai, B., Huang, L. and Xie, M. (2017) 'Bayesian Networks in Fault Diagnosis', *IEEE Transactions on Industrial Informatics*. doi: 10.1109/TII.2017.2695583.

CH Cheng, CP Chan, Y. S. (2019) ‘A novel purity-based k nearest neighbors imputation method and its application in financial distress prediction’, *Engineering Applications of Artificial Intelligence*, 81, pp. 283–299.

Chen, J. *et al.* (2019) ‘Gated recurrent unit based recurrent neural network for remaining useful life prediction of nonlinear deterioration process’, *Reliability Engineering and System Safety*. doi: 10.1016/j.ress.2019.01.006.

Choi, E. *et al.* (2015) ‘Doctor AI: Predicting Clinical Events via Recurrent Neural Networks’, *Proceedings of Machine Learning for Healthcare 2016 JMLR W&C Track*. doi: 10.1002/aur.1474.Replication.

Clark, T. G. and Altman, D. G. (2003) ‘Developing a prognostic model in the presence of missing data: An ovarian cancer case study’, *Journal of Clinical Epidemiology*. doi: 10.1016/S0895-4356(02)00539-5.

Das, S. and Suganthan, P. N. (2011) ‘Differential evolution: A survey of the state-of-the-art’, *IEEE Transactions on Evolutionary Computation*, 15(1), pp. 4–31. doi: 10.1109/TEVC.2010.2059031.

Donders, A. R. T. *et al.* (2006) ‘Review: A gentle introduction to imputation of missing values’, *Journal of Clinical Epidemiology*. doi: 10.1016/j.jclinepi.2006.01.014.

Eekhout, I. *et al.* (2012) ‘Missing data: a systematic review of how they are reported and handled.’, *Epidemiology (Cambridge, Mass.)*. doi: 10.1097/EDE.0b013e3182576cdb.

Fink, O., Zio, E. and Weidmann, U. (2015) ‘A Classification Framework for Predicting Components’ Remaining Useful Life Based on Discrete-Event Diagnostic Data’, *IEEE Transactions on Reliability*. doi: 10.1109/TR.2015.2440531.

Friedman, J. H. (2002) ‘Stochastic gradient boosting’, *Computational Statistics and Data Analysis*. doi: 10.1016/S0167-9473(01)00065-2.

Geraci, M. V. and Gnabo, J. Y. (2018) ‘Measuring Interconnectedness between Financial Institutions with Bayesian Time-Varying Vector Autoregressions’, *Journal of Financial and Quantitative Analysis*. doi: 10.1017/S0022109018000108.

Goudarzi, A., Shabani, A. and Stefanovic, D. (2015) ‘Product reservoir computing: Time-series computation with multiplicative neurons’, in *Proceedings of the International Joint Conference on Neural Networks*. doi: 10.1109/IJCNN.2015.7280453.

Gugulothu, N. *et al.* (2017) ‘Predicting Remaining Useful Life using Time Series Embeddings based on Recurrent Neural Networks’, *Journal of learning disabilities*. doi: 10.1177/002221947901200611.

Guo, L. *et al.* (2017) ‘A recurrent neural network based health indicator for remaining useful life prediction of bearings’, *Neurocomputing*, 240, pp. 98–109. doi: 10.1016/j.neucom.2017.02.045.

Hauskrecht, M. (2004) ‘Ensemble methods . Bagging and Boosting Bagging (Bootstrap Aggregating)’, *Machine Learning*.

Heidari Bafroui, H. and Ohadi, A. (2014) ‘Application of wavelet energy and Shannon entropy for feature extraction in gearbox fault detection under varying speed conditions’, *Neurocomputing*, 133, pp. 437–445. doi: 10.1016/j.neucom.2013.12.018.

Honaker, J. and King, G. (2010) ‘What to do about missing values in time-series cross-section data’, *American Journal of Political Science*, 54(2), pp. 561–581. doi: 10.1111/j.1540-5907.2010.00447.x.

Hu, C. *et al.* (2012) ‘Ensemble of data-driven prognostic algorithms for robust prediction of remaining useful life’, *Reliability Engineering and System Safety*. doi: 10.1016/j.ress.2012.03.008.

Hu, C., Youn, B. D. and Kim, T. (2012) ‘Semi-supervised learning with co-training for data-driven prognostics’, in *PHM 2012 - 2012 IEEE Int. Conf.on Prognostics and Health Management: Enhancing Safety, Efficiency, Availability, and Effectiveness of Systems Through PHM Technology and Application, Conference Program*. doi: 10.1109/ICPHM.2012.6299526.

Huang, C. G., Huang, H. Z. and Li, Y. F. (2019) ‘A Bidirectional LSTM Prognostics Method Under Multiple Operational Conditions’, *IEEE Transactions on Industrial Electronics*. doi: 10.1109/TIE.2019.2891463.

Hwang, J. T. G. and Ding, A. A. (1997) ‘Prediction Intervals for Artificial Neural Networks’, *Journal of the American Statistical Association*. doi: 10.1080/01621459.1997.10474027.

Inubushi, M. and Yoshimura, K. (2017) ‘Reservoir Computing beyond Memory-Nonlinearity Trade-off’, *Scientific Reports*, 7(1). doi: 10.1038/s41598-017-10257-6.

Jaeger, H. (2004) ‘Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication’, *Science*, 304(5667), pp. 78–80. doi: 10.1126/science.1091277.

Jaeger, H. (2005) ‘A tutorial on training recurrent neural networks , covering BPPT , RTRL , EKF and the " echo state network " approach’, *ReVision*, 2002, pp. 1–46. Available at: http://www.mendeley.com/catalog/tutorial-training-recurrent-neural-networks-covering-bppt-rtrl-ekf-echo-state-network-approach/.

Jaeger, H. *et al.* (2007) ‘Optimization and applications of echo state networks with leaky- integrator neurons’, *Neural Networks*, 20(3), pp. 335–352. doi: 10.1016/j.neunet.2007.04.016.

Jaeger, H. (2010) ‘The “ echo state ” approach to analysing and training recurrent neural networks – with an Erratum note 1’, *GMD Report*, (148), pp. 1–47. doi: citeulike-article-id:9635932.

Khelif, R. *et al.* (2017) ‘Direct Remaining Useful Life Estimation Based on Support Vector Regression’, *IEEE Transactions on Industrial Electronics*. doi: 10.1109/TIE.2016.2623260.

Khosravi, A. *et al.* (2011a) ‘Comprehensive review of neural network-based prediction intervals and new advances’, *IEEE Transactions on Neural Networks*. doi: 10.1109/TNN.2011.2162110.

Khosravi, A. *et al.* (2011b) 'Lower upper bound estimation method for construction of neural network-based prediction intervals', *IEEE Transactions on Neural Networks*. doi: 10.1109/TNN.2010.2096824.

Kim, Jihyun *et al.* (2016) 'Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection', in *2016 International Conference on Platform Technology and Service, PlatCon 2016 - Proceedings*. doi: 10.1109/PlatCon.2016.7456805.

KP, S. (2019) 'RNNSecureNet: Recurrent neural networks for Cyber security use-cases', *arXiv preprint arXiv:1901.04281*.

Lamb, A. *et al.* (2016) 'Professor Forcing: A New Algorithm for Training Recurrent Networks', (Nips), pp. 1–9. doi: 10.1109/CVPR.2015.7299068.

Lei, Y. *et al.* (2018) 'Machinery health prognostics: A systematic review from data acquisition to RUL prediction', *Mechanical Systems and Signal Processing*. doi: 10.1016/j.ymssp.2017.11.016.

Leturiondo, U. *et al.* (2017) 'Architecture for hybrid modelling and its application to diagnosis and prognosis with missing data', *Measurement: Journal of the International Measurement Confederation*. doi: 10.1016/j.measurement.2017.02.003.

Liao, L. and Köttig, F. (2014) 'Review of hybrid prognostics approaches for remaining useful life prediction of engineered systems, and an application to battery life prediction', *IEEE Transactions on Reliability*. doi: 10.1109/TR.2014.2299152.

Lim, P. *et al.* (2017) 'Multimodal degradation prognostics based on switching kalman filter ensemble', *IEEE Transactions on Neural Networks and Learning Systems*. doi: 10.1109/TNNLS.2015.2504389.

Little, R. J. a and Rubin, D. B. (2002) *Statistical Analysis with Missing Data*, *Statistical analysis with missing data Second edition*. doi: 10.2307/1533221.

Liu, J., Li, Y. F. and Zio, E. (2017) 'A SVM framework for fault detection of the braking system in a high speed train', *Mechanical Systems and Signal Processing*. Elsevier, 87(October 2016), pp. 401–409. doi: 10.1016/j.ymssp.2016.10.034.

Lukoševičius, M. and Jaeger, H. (2009) 'Reservoir computing approaches to recurrent neural network training', *Computer Science Review*, 3(3), pp. 127–149. doi: 10.1016/j.cosrev.2009.03.005.

Malhi, A., Yan, R. and Gao, R. X. (2011) 'Prognosis of defect propagation based on recurrent neural networks', *IEEE Transactions on Instrumentation and Measurement*. doi: 10.1109/TIM.2010.2078296.

Marshall, A. *et al.* (2010) 'Comparison of techniques for handling missing covariate data within prognostic modelling studies: A simulation study', *BMC Medical Research Methodology*. doi: 10.1186/1471-2288-10-7.

Medjaher, K., Tobon-Mejia, D. a and Zerhouni, N. (2012) 'Remaining Useful Life Estimation of Critical Components With Application to Bearings', *IEEE Transactions on Reliability*, 61(2), pp. 292–302. doi: 10.1109/TR.2012.2194175.

Mendes-Moreira, J. *et al.* (2012) 'Ensemble approaches for regression: A survey', *ACM Computing Surveys*. doi: 10.1145/2379776.2379786.

Morando, S. *et al.* (2015) 'ANOVA method applied to proton exchange membrane fuel cell ageing forecasting using an echo state network', *Mathematics and Computers in Simulation*, 131, p. doi: 10.1016/j.matcom.2015.06.009.

Morando, S. *et al.* (2017) 'Proton exchange membrane fuel cell ageing forecasting algorithm based on Echo State Network', *International Journal of Hydrogen Energy*. doi: 10.1016/j.ijhydene.2016.05.286.

Moustapha, a. I. and Selmic, R. R. (2008) 'Wireless Sensor Network Modeling Using Modified Recurrent Neural Networks: Application to Fault Detection', *IEEE Transactions on Instrumentation and Measurement*, 57(5), pp. 981–988. doi: 10.1109/TIM.2007.913803.

Palacios, A. *et al.* (2015) 'Sequential pattern mining applied to aeroengine condition monitoring with uncertain health data', *Engineering Applications of Artificial Intelligence*. doi: 10.1016/j.engappai.2015.05.003.

Perisse, F. *et al.* (2006) 'Simple model of an electrolytic capacitor taking into account the temperature and aging time', *Electrical Engineering*. doi: 10.1007/s00202-004-0265-z.

Prytz, R. *et al.* (2015) 'Predicting the need for vehicle compressor repairs using maintenance records and logged vehicle data', *Engineering Applications of Artificial Intelligence*. doi: 10.1016/j.engappai.2015.02.009.

Qiao, J. *et al.* (2016) 'Growing Echo-State Network With Multiple Subreservoirs', *IEEE Transactions on Neural Networks and Learning Systems*, 28(2), pp. 391–404. doi: 10.1109/TNNLS.2016.2514275.

Ranjbar, M. *et al.* (2015) 'An imputation-based matrix factorization method for improving accuracy of collaborative filtering systems', *Engineering Applications of Artificial Intelligence*. doi: 10.1016/j.engappai.2015.08.010.

Razavi-Far, R. *et al.* (2019) 'An integrated imputation-prediction scheme for prognostics of battery data with missing observations', *Expert Systems with Applications*. doi: 10.1016/j.eswa.2018.08.033.

Rekik, M., Gransart, C. and Berbineau, M. (2018) 'Cyber-physical security risk assessment for train control and monitoring systems', in *2018 IEEE Conference on Communications and Network Security, CNS 2018*. doi: 10.1109/CNS.2018.8433201.

Rigamonti, M., Baraldi, P., Zio, E., Roychoudhury, I., *et al.* (2016) 'Echo State Network for the Remaining Useful Life Prediction of a Turbofan Engine', in *EUROPEAN CONFERENCE OF THE PROGNOSTICS AND HEALTH MANAGEMENT SOCIETY 2016*, pp. 1–15.

Rigamonti, M., Baraldi, P., Zio, E., Astigarraga, D., *et al.* (2016) 'Particle Filter-Based Prognostics for an Electrolytic Capacitor Working in Variable Operating Conditions', *IEEE Transactions on Power Electronics*.

doi: 10.1109/TPEL.2015.2418198.

Rigamonti, M. *et al.* (2017) 'Ensemble of optimized echo state networks for remaining useful life prediction', *Neurocomputing*, 0, pp. 1–18. doi: 10.1016/j.neucom.2017.11.062.

Rosenberg, I. *et al.* (2017) 'Generic Black-Box End-to-End Attack against RNNs and Other {API} Calls Based Malware Classifiers', *CoRR*.

Samanta, B. and Al-Balushi, K. R. (2003) 'Artificial neural network based fault diagnostics of rolling element bearings using time-domain features', *Mechanical Systems and Signal Processing*, 17(2), pp. 317–328. doi: 10.1006/mssp.2001.1462.

Sardá-Espinosa, A., Subbiah, S. and Bartz-Beielstein, T. (2017) 'Conditional inference trees for knowledge extraction from motor health condition data', *Engineering Applications of Artificial Intelligence*. doi: 10.1016/j.engappai.2017.03.008.

Saxena, A. *et al.* (2010) 'Metrics for Offline Evaluation of Prognostic Performance', *International Journal of Prognostics and Health Management*, (1), pp. 1–20. Available at: http://72.27.231.73/sites/phmsociety.org/files/phm_submission/2010/ijPHM_10_001.pdf.

Schafer, J. L. (1999) 'Multiple imputation: A primer', *Statistical Methods in Medical Research*. doi: 10.1191/096228099671525676.

Schafer, J. L. and Graham, J. W. (2002) 'Missing data: Our view of the state of the art', *Psychological Methods*. doi: 10.1037/1082-989X.7.2.147.

Schwabacher, M. and Goebel, K. (2007) 'A Survey of Artificial Intelligence for Prognostics', in *AAAI Fall Symposium*.

Teoh, T. T. *et al.* (2018) 'Applying RNN and J48 Deep Learning in Android Cyber Security Space for Threat Analysis', in *2018 International Conference on Smart Computing and Electronic Enterprise, ICSCEE 2018*. doi: 10.1109/ICSCEE.2018.8538405.

Tobiyama, S. *et al.* (2016) 'Malware Detection with Deep Neural Network Using Process Behavior', in *Proceedings - International Computer Software and Applications Conference*. doi: 10.1109/COMPSAC.2016.151.

Tse, P. W. and Atherton, D. P. (1999) 'Prediction of Machine Deterioration Using Vibration Based Fault Trends and Recurrent Neural Networks', *Journal of Vibration and Acoustics*, 121(3), p. 355. doi: 10.1115/1.2893988.

Tsividis, Y. (2010) 'Event-driven data acquisition and digital signal processing-A tutorial', *IEEE Transactions on Circuits and Systems II: Express Briefs*, 57(8), pp. 577–581. doi: 10.1109/TCSII.2010.2056012.

Tuor, A. *et al.* (2017) 'Deep learning for unsupervised insider threat detection in structured cybersecurity data streams', in *AAAI Workshop - Technical Report*.

De Veaux, R. D. *et al.* (1998) 'Prediction intervals for neural networks via nonlinear regression', *Technometrics*. doi: 10.1080/00401706.1998.10485556.

Venet, P., Darnand, H. and Grellet, G. (1993) 'Detection of faults of filter capacitors in a converter. Application to predictive maintenance', in *INTELEC, International Telecommunications Energy Conference (Proceedings)*. doi: 10.1109/INTLEC.1993.388568.

Vergouw, D. *et al.* (2010) 'The search for stable prognostic models in multiple imputed data sets', *BMC Medical Research Methodology*. doi: 10.1186/1471-2288-10-81.

Vinayakumar, R., Soman, K. P. and Poornachandrany, P. (2017) 'Applying convolutional neural network for network intrusion detection', in *2017 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2017*. doi: 10.1109/ICACCI.2017.8126009.

Weijters, a. J. M. M. and van der Aalst, W. M. P. (2003) 'Rediscovering Workflow Models from Event-Based Data using Little Thumb', *Integrated ComputerAided Engineering*, 10, pp. 151–162. Available at: http://iospress.metapress.com/index/8PUQ22EUMRVA7VYP.pdf.

Williams, R. J. and Zipser, D. (1989) 'A Learning Algorithm for Continually Running Fully Recurrent Neural Networks', *Neural Computation*. doi: 10.1162/neco.1989.1.2.270.

Wootton, A. J., Day, C. R. and Haycock, P. W. (2015) 'An Echo State Network approach to structural health monitoring', in *Proceedings of the International Joint Conference on Neural Networks*. doi: 10.1109/IJCNN.2015.7280627.

Xiao, H. *et al.* (2017) 'Fault diagnosis and prognosis of wastewater processes with incomplete data by the auto-associative neural networks and ARMA model', *Chemometrics and Intelligent Laboratory Systems*. doi: 10.1016/j.chemolab.2016.12.009.

Xing, H.-J. and Liu, W.-T. (2019) 'Robust AdaBoost based ensemble of one-class support vector machines', *Information Fusion*. Elsevier B.V., 55(August 2019), pp. 45–58. doi: 10.1016/j.inffus.2019.08.002.

Yang, Z., Tang, K. and Yao, X. (2008) 'Self-adaptive differential evolution with neighborhood search', in *2008 IEEE Congress on Evolutionary Computation, CEC 2008*, pp. 1110–1116. doi: 10.1109/CEC.2008.4630935.

Yildiz, I. B., Jaeger, H. and Kiebel, S. J. (2012) 'Re-visiting the echo state property', *Neural Networks*, 35, pp. 1–9. doi: 10.1016/j.neunet.2012.07.005.

Yoon, K. and Hwang, C.-L. (1995) *Multiple attribute decision making: an introduction*, *Sage Publications Thousand Oaks CA*.

Zhang, Y. *et al.* (2018) 'Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries', *IEEE Transactions on Vehicular Technology*, 9545(c). doi: 10.1109/TVT.2018.2805189.

Zhong, S. *et al.* (2017) 'Genetic algorithm optimized double-reservoir echo state network for multi-regime time series prediction', *Neurocomputing*. doi: 10.1016/j.neucom.2017.01.053.

Zhou, H. *et al.* (2018) 'Echo state kernel recursive least squares algorithm for machine condition prediction', *Mechanical Systems and Signal Processing*. doi: 10.1016/j.ymssp.2018.03.047.

Zio, E. *et al.* (2012) 'Failure and reliability predictions by infinite impulse response locally recurrent neural networks', *Chemical Engineering Transactions*. doi: 10.3303/CET1226020.

Zio, E. (2012) 'Prognostics and Health Management of Industrial Equipment', *Diagnostics and Prognostics of Engineering Systems: Methods and Techniques*, pp. 333–356. doi: 10.4018/978-1-4666-2095-7.

Zio, E., Broggi, M. and Pedroni, N. (2009) 'Nuclear reactor dynamics on-line estimation by Locally Recurrent Neural Networks', *Progress in Nuclear Energy*. doi: 10.1016/j.pnucene.2008.11.006.

Zio, E. and Di Maio, F. (2010) 'A data-driven fuzzy approach for predicting the remaining useful life in dynamic failure scenarios of a nuclear system', *Reliability Engineering and System Safety*, 95(1), pp. 49–57. doi: 10.1016/j.ress.2009.08.001.