A long-term prediction approach based on long short-term memory neural networks with automatic parameter optimization by Tree-structured Parzen Estimator and applied to time-series data of NPP steam generators

Hoang-Phuong Nguyen, Jie Liu, Enrico Zio

A long-term prediction approach based on Long Short-Term Memory neural network with automatic parameter optimization by Tree-structured Parzen Estimator and applied to time-series data of NPP steam generators

Hoang-Phuong Nguyen[a], Jie Liu[b], Enrico Zio[c,d,e*]

Affiliation:

[a] Chair on System Science and the Energetic Challenge, CentraleSupélec, Université Paris-Saclay, France

[b] School of Reliability and System Engineering, Beihang University, China

[c] MINES ParisTech / PSL Université Paris, Centre de Recherche sur les Risques et les Crises (CRC), France

[d] Department of Energy, Politecnico di Milano, Milano, Italy

[e] Eminent Scholar, Department of Nuclear Engineering, Kyung Hee University, South Korea

Address:

[a] 9 rue Joliot-Curie, 91192 Gif-sur-Yvette, France

[b] 37 Xueyuan Road, Haidian, Beijing, China

[c] 1 rue Claude Daunesse, 06904 Sophia Antipolis, France

[d] Via La Masa 34, 20156 Milano, Italy

[e] 26 Kyungheedae-ro, Hoegi-dong, Dongdaemun-gu, Seoul, South Korea

Email:

[1] hoang-phuong.nguyen@centralesupelec.fr

[2] liujie805@buaa.edu.cn

[3] enrico.zio@mines-paristech.fr, enrico.zio@polimi.it

Corresponding author:

[*] Enrico Zio, MINES ParisTech / PSL Université Paris, Centre de Recherche sur les Risques et les Crises (CRC), France

Email: enrico.zio@mines-paristech.fr, enrico.zio@polimi.it

Abstract

Developing an accurate and reliable multi-step ahead prediction model is a key problem in many Prognostics and Health Management (PHM) applications. Inevitably, the further one attempts to predict into the future, the harder it is to achieve an accurate and stable prediction due to increasing uncertainty and error accumulation. In this paper, we address this problem by proposing a prediction model based on Long Short-Term Memory (LSTM), a deep neural network developed for dealing with the long-term dependencies in time-series data. Our proposed prediction model also tackles two additional issues. Firstly, the hyperparameters of the proposed model are automatically tuned by a Bayesian optimization algorithm, called Tree-structured Parzen Estimator (TPE). Secondly, the proposed model allows assessing the uncertainty on the prediction. To validate the performance of the proposed model, a case study considering steam generator data acquired from different French nuclear power plants (NPPs) is carried out. Alternative prediction models are also considered for comparison purposes.

Keywords

prognostics and health management, time-series forecasting, multi-step ahead prediction, long-short term memory, nuclear power plant prognostics, steam generator.

Nomenclature

Abbreviations

AI	Artificial intelligence
ANN	Artificial neural network
ARMA	Autoregressive moving average
ARIMA	Autoregressive integrated moving average
BO	Bayesian optimization
CNN	Convolutional neural network
DBN	Deep belief network
EDF	Électricité de France
EI	Expected improvement
FNN	False nearest neighbor
FIS	Fuzzy interference system
LSTM	Long short-term memory
MAPE	Mean absolute percentage error
MASE	Mean absolute scaled error
MC	Monte Carlo
MIMO	Multi-input multi-output
MLP-MIMO	Multi-output multilayer perceptron neural network using MIMO strategy
MLP-REC	Single-input multilayer perceptron neural network using recursive strategy
MSE	Mean square error
NF	Neuro-fuzzy
NPP	Nuclear power plant
PHM	Prognostics and health management
PWR	Pressurized water reactor
RBM	Restricted Boltzmann machine
RMSE	Root mean square error
RNN	Recurrent neural network
RS	Random search
RUL	Remaining useful life
SG	Steam generator
SVM	Support vector machine
SVR-MIMO	Multi-Input Multi-Output support vector regression using MIMO strategy
SVR-REC	Single-input support vector regression using recursive strategy
TPE	Tree-structured Parzen estimator
WRL	Wide range level

Symbols

C_t	output of the LSTM cell state at time t
$\tilde{C}_t$	potential values of the LSTM cell state at time t
d	embedding dimension
f	prediction model
f_R	one-step ahead prediction model
$f_{D,h}$	direct prediction model for the horizon time t+h
f_{MIMO}	MIMO prediction model
f_t	output of the LSTM forget gate at time t
h_t	output of the repeating network module of a LSTM at time t
H	prediction horizon
i_t	output of the LSTM input gate at time t
L	number of hidden layers in the LSTM network
max_iter	number of optimization iterations
N_{init}	number of TPE startup iterations
N_{MC}	number of MC dropout realizations
o_t	output of the LSTM output gate at time t
$\Pr_{bad}$	probability of being in the bad group in the TPE algorithm
$\Pr_{good}$	probability of being in the good group in the TPE algorithm
$r^{(l)}$	vector of independent Bernoulli random variables at layer l of the LSTM network
t	time t
x_t	observed value at time t
$\hat{x}_t$	predicted value at time t
$y^{(l)}$	output vector of the hidden layer l of the LSTM network
$\tilde{y}^{(l)}$	thinned output vector of the hidden layer l obtained by using dropout
$z^{(l)}$	input vector of the hidden layer l of the LSTM network
(W_C, b_C)	weight and bias of the LSTM cell state, respectively
(W_f, b_f)	weight and bias of the LSTM forget gate, respectively
(W_i, b_i)	weight and bias of the LSTM input gate, respectively
(W_o, b_o)	weight and bias of the LSTM output gate, respectively
σ	a network layer in the repeating network module of a LSTM

θ	hyperparameter set
θ^*	selected hyperparameter set

1 Introduction

In recent years, prognostics and health management (PHM) has attracted increasing attention from academic researchers and practitioners in different industrial sectors. The primary characteristic of PHM is that it can enable estimation and prediction of the health state of components and systems, by making use of past, present and future knowledge, information and data on their operations, and this capability can be used to identify malfunctions and anticipate failure patterns [1]. This allows estimating the remaining useful life (RUL) of components and systems, and scheduling the maintenance interventions for the most opportune and convenient instances. By so doing, the availability and reliability of the assets can be maximized, with reduced unscheduled shutdowns and maintenance costs.

Developing models for efficient PHM is a challenging task, with several issues to be addressed. Among them, determining an appropriate horizon for the prediction, i.e. how far into the future the model should predict and with what accuracy, is crucial and application dependent, in the sense that it depends on the use that is made of the prediction, typically for taking some decisions [2]. For instance, the selected horizon should be suitably long to allow that maintenance actions be timely carried out. This often requires long-term predictions in practice. However, long-term predictions are known to suffer from increasing uncertainties, which may arise from the accumulation of prediction errors or from the complex interactions and correlations in the underlying process at different time steps. This has challenged and somewhat limited the research on long-term prognostics for many years [3], [4]. To address this problem, the main focus of this paper is the development of a prognostic framework for the long-term prediction of parameters relevant to the operation of the steam generators (SGs) in nuclear power plants (NPPs).

Depending on the information and data available for the model development, prognostic approaches can be divided into two main categories: model-based and data-driven approaches [5]. Model-based approaches predict the degradation evolution by formalizing it into physical analytical or

computational models. These approaches are used in applications where the model of the degradation process exists and is not too complicated, e.g. models of fatigue crack growth [6], [7], of capacity degradation in Lithium-ion batteries [8], [9]. Alternatively, data-driven approaches utilize condition monitoring data collected from sensors to learn and predict the component or system behavior and degradation via statistical and artificial intelligent (AI) models, such as autoregressive integrated moving average (ARIMA) [10], artificial neural network (ANN) [11]–[14], neuro-fuzzy (NF) [2] and support vector machine (SVM) [15]–[17]. Due to the data-adaptive nature, data-driven approaches are quite appropriate for prognostic real-world applications where models are not available whereas obtaining condition monitoring data is becoming convenient with smart sensors.

When applying data-driven approaches to prediction, models like ANN, NF and SVM are usually limited in extracting and utilizing the temporal information of the given data which is necessary for prediction purposes. More specifically, these approaches consider each time step independently and make the prediction as a static mapping, which often takes into account only the current state of the process [18]. Recently, a connectionist neural network model called recurrent neural network (RNN) has been proposed to account for the dynamics [19]. RNN is a network with feedback connections from the hidden and output layers to the preceding ones, by which the dynamics of sequential data can be captured and the memories of the previous patterns are retained via cycles in the network. In the last decade, RNNs have been extensively investigated for a variety of prognostic applications, including engine systems [20]–[23], lithium-ion batteries [24]–[26], rolling element bearings [27]–[30] and fuel cells [31], [32]. Zhang et al. [24] utilized a RNN to extract the long-term dependencies underlying in the battery capacity degradation process. The obtained results showed that RNN outperformed the classical data-driven models in prediction robustness and accuracy. In [30], the RNN model was modified with an incremental learning technique, which was then applied for predicting the long-term propagation of rolling element bearing degradation to failure. RNN has also been used in the construction of health indicators for generator bearings of wind turbines [33]. The obtained RNN-based health indicator showed its effectiveness for improving the prediction performance of the bearing RULs.

In this paper, a variational model of RNN, which is called long short-term memory (LSTM), is employed for developing a prognostic framework for SGs in NPPs. An important feature of the proposed framework is the ability to deal with a long-term prediction horizon. A multi-input multi-output (MIMO) prediction strategy and LSTM network are integrated to predict the equipment health conditions for multiple steps ahead. The proposed framework also handles two practical problems in prediction model development. On one hand, the performance of the prediction model depends on the time-series data acquired from sensors: any anomalous or missing data that can degrade the performance should be dealt with. On the other hand, an optimal model setting for different available datasets is crucial to successfully apply the prediction model to practical problems. The effective handling of these two issues is another contribution of this work.

In summary, the main contributions of this paper are as follows:

(1) A data preprocessing module consisting of an outlier removal and a missing data imputation methods is introduced for filtering and preparing the data for the prediction task.

(2) Automatic hyperparameter optimization based on the Tree-structured Parzen Estimator (TPE) algorithm is performed. The obtained results are compared to the conventional random search (RS) algorithm with respect to different data scenarios.

(3) Dropout regularization and Monte Carlo (MC) techniques are integrated to assess the prediction uncertainty of the proposed model.

(4) A case study using the data of SGs in French NPPs measured during the period 1992-2007 is carried out to evaluate the performance of the proposed LSTM-based framework for long-term prediction. To the authors' knowledge, this is the first study using LSTM for the long-term prediction and used on NPP SGs. Other prediction models are considered for comparison purposes.

The rest of the paper is organized as follows. Section 2 presents a brief introduction of time series prediction approaches. Section 3 introduces the LSTM neural network. The proposed multi-step ahead prediction model is presented in Section 4. Section 5 describes the experimental case study, and the obtained results and their discussion are presented in Section 6. Finally, Section 7 concludes the paper.

2 Time series prediction: background and related work

Time series is a sequence of observations collected over time from a particular measured variable of an engineering component or system. In general, the main objectives of time series analysis are: characterization, modeling and prediction (also called forecasting) [34]. Firstly, *characterization* aims to extract inherent structural characteristics of the measured variable, e.g. temporal trend, variance and seasonality. Then, the extracted information may be used to formulate an appropriate model for capturing long-term behavior of the system (*modeling*), or to estimate the evolution of the variable in the future *(prediction)*. This section presents a brief introduction of time series prediction approaches and further discusses the strategies for multi-step ahead prediction.

2.1 Time series prediction

The beginning of time series prediction might be set in 1927 when Yule [35] introduced the first autoregressive technique for predicting the annual number of sunspots. In that original work, the prediction of the next time step was estimated as a weighted sum of previous observations of the time series. This idea has become the basis of data-driven approaches for time series prediction since then.

Among data-driven approaches, statistical models which attempt to express the future values as a linear function of the historical data have been popular and widely used in many applications of time series prediction, such as wind energy generation [36]–[39], weather forecasting [40]–[42], market demand forecasting [43], [44] and nuclear component prognostics [10]. Erdem and Shi [36] used an autoregressive moving average (ARMA) for predicting wind speed and direction. Kavasseri and Seetharamen [38] proposed a variant model of ARIMA which was called fractional-ARIMA in order to extract the long dependency features of the time series data and enhance the prediction accuracy over long-term horizons. In a nuclear application, Nguyen et al. [10] applied an ARIMA model to predict the long-term evolution of the tube supporting plate clogging degradation of NPP SGs for the first time, in which the predictions were performed up to 3 months ahead.

Although statistical models have shown their notable prediction accuracy and flexibility in different

time series applications, one of their major drawbacks is the presumed linear form of the associated data, which has limited their applicability to many modern dynamic systems where the collected data are usually nonlinear and non-stationary [45], [46]. To address this problem, several machine learning algorithms have been employed in the time series prediction area, such as SVM [47]–[49], ANNs [50]–[53] and fuzzy interference system (FIS) [54], [55]. Unlike statistical approaches, machine learning models can automatically learn arbitrary complex mappings between inputs and outputs directly from the historical data and perform accurate predictions without any assumption about the mapping functions required. In addition, another advantage of machine learning approaches is the recently rapid advancements of information science technologies, particularly Big Data and deep learning techniques, which are offering opportunities for new developments in time series analysis. Kuremoto et al. [56] proposed a deep belief network (DBN) with restricted Boltzmann machines (RBMs) to address problems of initialization and local optima in chaotic time series forecasting, which was shown to outperform conventional shallow learning models. Wang et al. [57] presented a prediction model for probabilistic wind power prediction based on a convolutional neural network (CNN) model to automatically extract deep invariant structures and hidden nonlinear features exhibited at separated frequency bands of the data. A specialized kind of deep neural networks proposed for sequential data analysis is RNNs, which aim to capture the dynamics of sequential data and be able to retain the memories of the previous patterns via cycles of feedback connections between the network layers [19]. Wang and Li [46] presented a hybrid model integrating a RNN model and an optimal feature extraction technique for multi-step ahead wind speed prediction. Likewise, Li et al. [58] utilized LSTM RNN for predicting 5 steps ahead of the wind speed time series.

2.2 Multi-step ahead prediction strategies

Given a univariate time series of the observations collected up to time t, $\{x_1, x_2, ..., x_t\}$, the main goal is to predict the H next observations $\{\hat{x}_{t+h}\}, h \in [1, H]$, which can be formulated as below:

$$\{\hat{x}_{t+1}, \hat{x}_{t+2}, ..., \hat{x}_{t+H}\} = f(x_t, x_{t-1}, ..., x_{t-d+1}), \quad (1)$$

where f is the prediction model and d is the embedding dimension (or the number of lagged values).

Depending on the desired horizon H, a prediction method can be classified into short-, medium-, or long-term prediction. As aforementioned, the further in the future one attempts to predict, the harder it is to achieve an accurate prediction due to the increasing uncertainty and accumulation of errors. To address this problem, there are three popular prediction strategies, namely recursive, direct and MIMO predictions, which are described as follows [2].

2.2.1 Recursive prediction strategy

The recursive strategy attempts to train a model focused solely on one-step ahead prediction:

$$\hat{x}_{t+1} = f_R(x_t, x_{t-1}, ..., x_{t-d+1}) \tag{2}$$

where f_R is the one-step ahead prediction model.

After the model is trained, the predictions are recursively estimated. In other words, intermediate predictions are used as inputs for predicting next values until the prediction at the time horizon H, $\hat{x}_{t+H}$, is obtained:

$$\begin{aligned} \hat{x}_{t+1} &= f_R(x_t, x_{t-1}, ..., x_{t-d+1}) \\ \hat{x}_{t+2} &= f_R(\hat{x}_{t+1}, x_t, ..., x_{t-d+2}) \\ &\vdots \qquad\qquad \vdots \\ \hat{x}_{t+H} &= f_R(\hat{x}_{t+H-1}, \hat{x}_{t+H-2}, ..., \hat{x}_{t+H-d+1}) \end{aligned} \tag{3}$$

An advantage of the recursive strategy is its low computational cost since only one single model is required for training. However, the prediction errors of the previous steps can easily accumulate in the next predictions, resulting in the decrease of accuracy in the long run. Besides, this prediction strategy does not take into account the data dependencies among time steps.

2.2.2 Direct prediction strategy

In contrast to the recursive strategy which uses a single model, the direct strategy [59] constructs a set of H different models for different time steps and the same input data are used for feeding all the models as below:

$$\begin{aligned} \hat{x}_{t+1} &= f_{D,1}(x_t, x_{t-1}, ..., x_{t-d+1}) \\ \hat{x}_{t+2} &= f_{D,2}(x_t, x_{t-1}, ..., x_{t-d+1}) \\ &\vdots \qquad\qquad \vdots \\ \hat{x}_{t+H} &= f_{D,H}(x_t, x_{t-1}, ..., x_{t-d+1}) \end{aligned} \tag{4}$$

where $f_{D,h}$ is the direct prediction model tuned to perform the prediction $\hat{x}_{t+h}$ at time $t+h, h \in [1, H]$.

In the direct strategy, each prediction model is trained and dedicated to a certain horizon, so the error accumulation can be avoided. However, training different prediction models will greatly increase the prediction complexity and time consumption, and, like the recursive strategy, the direct strategy does not take into account the dependencies among time-series observations.

2.2.3 MIMO prediction strategy

Unlike the recursive and direct approaches, the MIMO approach is a multiple output strategy, in which the output of the prediction model is a vector of future values predicted by using only one model [60]:

$$\{\hat{x}_{t+1}, \hat{x}_{t+2}, ..., \hat{x}_{t+H}\} = f_{MIMO}(x_t, x_{t-1}, ..., x_{t-d+1}) \tag{5}$$

where f_{MIMO} is the multiple output prediction model. In this sense, the objective function during the model training is to simultaneously minimize the prediction errors on different horizons. By so doing, the MIMO strategy can preserve the temporal stochastic dependencies of sequential data, addressing the limitation of the recursive and direct approaches. On the other hand, the computational cost of the MIMO approach is less than that of the direct approach because it requires only one model to be trained.

3 Long short-term memory (LSTM)

3.1 Network architecture

LSTM is a variant of RNNs developed for dealing with the long-term dependency problem, which is also known as “vanishing gradients” or “exploding gradients” problem [61]. In general, an LSTM consists of a chain of repeating network modules, in which each module contains four interacting layers, as illustrated in Fig. 1 [62].

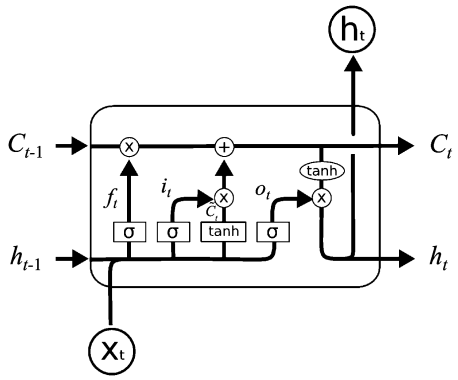


Fig. 1. The schematic of a repeating network module in a LSTM network.

The key element of a LSTM network is the cell state *C*, which is depicted as the horizontal line running through the top of the diagram in Fig. 1. This cell state plays a role as a network memory, where information is added or removed via regulated structures called gates, which can optionally let information through. They are composed of a sigmoid neural network layer and a pointwise multiplication operation. An LSTM consists of three gates, including forget, input and output gates, in order to protect and control the cell state. Details on the LSTM procedure are described as follows.

At time *t*, an input x_t is fed to the network. The forget gate first decides which information from the previous output h_{t-1} is discarded or kept, and then the output of the forget gate is calculated as:

$$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right] + b_f\right), \tag{6}$$

where (W_f, b_f) are the input weights and bias of the forget gate, respectively, σ is a nonlinear function (e.g. sigmoid function) and " . " means matrix multiplication.

The next step is to determine which new information will be stored in the cell state, leading to the two following calculations. First, the input gate decides which states will be updated; then, a *tanh* layer generates a vector of new values $\tilde{C}_t$ that could be added to the cell state, as follows:

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right), \tag{7}$$

$$\tilde{C}_t = \tanh\left(W_C \cdot \left[h_{t-1}, x_t\right] + b_C\right), \tag{8}$$

where (W_i, b_i) and (W_C, b_C) are the input weights and bias of the input gate and the cell state layer, respectively. The outputs obtained from the forget gate, input gate and *tanh* layer are, then, used to

update the new cell state C_t:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t. \tag{9}$$

Finally, the network output h_t is generated by the output gate and a *tanh* function, as:

$$o_t = \sigma\left(W_o \cdot \left[h_{t-1}, x_t\right] + b_o\right), \tag{10}$$

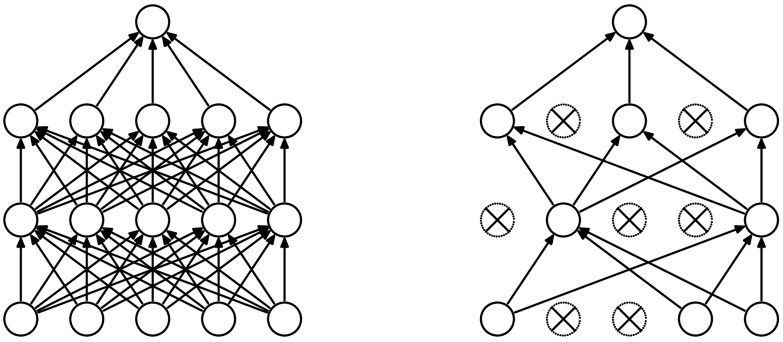
$$h_t = o_t * \tanh(C_t), \tag{11}$$

where (W_o, b_o) are the input weights and bias of the output gate, respectively.

3.2 Dropout regularization

A well-known and critical problem of deep neural networks such as LSTM is overfitting [63]. That is, when the training data is limited, complicated mappings between the inputs and outputs that are learned by the network might be the result of sampling noise, which only exist in the training set but not in the real test set. One way to regularize such a network is averaging the outputs of all possible configurations of the parameters, in which each configuration is weighted by its posterior probability given by the training data [64]. This method can be applied only for simple or small networks. With large neural networks, the computation for training many different network architectures or training one architecture on different data sets is very expensive. Dropout is a technique that addresses this issue [64].

A motivation for dropout comes from a theory of sexual reproduction [65], in which new genes are naturally selected to spread throughout the population based on their competitiveness and less co-adaptation which may reduce the chance of a new gene improving the fitness of an individual. Likewise, dropout aims to train each hidden unit in a neural network with a randomly chosen sample of other units. By dropping a unit out, we temporarily remove it from the network along with all its connections during the training process as illustrated in Fig. 2, in order to prevent units from high co-adaptation. By so doing, each hidden unit becomes more robust and is able to create useful features on its own without relying on other units, which helps the network avoid overfitting.



(a) (b)

Fig. 2. An example of a dropout network model [64]: (a) A fully connected 2-hidden layers network; (b) The network obtained by applying dropout. Crossed units are excluded.

Consider a neural network with L hidden layers, in which the input and output vectors of layer l (for $l \in \{1, \ldots, L\}$) are denoted as $z^{(l)}$ and $y^{(l)}$, respectively. $w^{(l)}$ and $b^{(l)}$ are the weights and biases of layer l, respectively. For a standard neural network, the feed-forward operation can be described as:

$$z_i^{(l+1)} = w_i^{(l+1)} y^l + b_i^{(l+1)}, \tag{12}$$

$$y_i^{(l+1)} = f\left(z_i^{(l+1)}\right), \tag{13}$$

where f is the activation function and i denotes the index of hidden unit, as illustrated in Fig. 3(a).

With a dropout network (Fig. 3(b)), a vector of independent Bernoulli random variables $r^{(l)}$ with probability p is used at each hidden layer l to generate the thinned outputs $\tilde{y}^{(l)}$ as follows:

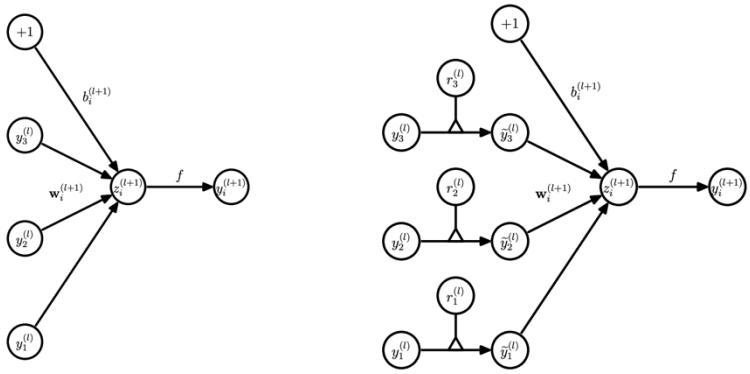
$$r_j^{(l)} \sim Bernoulli(p), \tag{14}$$

$$\tilde{y}^{(l)} = r^{(l)} * y^{(l)}, \tag{15}$$

where * denotes an element-wise product. The thinned outputs are, then, used as inputs to the next layer of the feed-forward operation:

$$z_i^{(l+1)} = w_i^{(l+1)} \tilde{y}^{(l)} + b_i^{(l+1)}, \tag{16}$$

$$y_i^{(l+1)} = f\left(z_i^{(l+1)}\right), \tag{17}$$



(a) Standard network (b) Dropout network

Fig. 3. Comparison of the basic operations of a standard and dropout network [64].

The dropout technique was shown to significantly reduce overfitting and improve the performance of standard neural networks in a wide variety of application domains, including handwriting recognition, speech recognition, image processing, object classification and computational biology [64]. In this paper, dropout is used in the input and hidden layers of the proposed LSTM model in order to prevent overfitting and quantify the uncertainty information of the multi-step ahead predictions, which is further described in Section 4.3.

4 Proposed LSTM-based prognostic framework

In this section, we present a prognostic framework for the multi-step ahead prediction of the time-series data from SGs, as illustrated in Fig. 4.

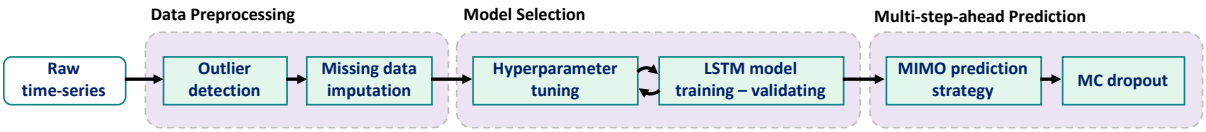


Fig. 4. The flowchart of the proposed multi-step ahead prediction framework for SGs.

The proposed framework consists of three main stages: data preprocessing, model selection and multi-step ahead prediction. Firstly, the data preprocessing stage is responsible for preparing the data for training and testing the prediction model. Then, in the second stage, a LSTM-based model is built

for the MIMO prediction using the training data and its hyperparameters are automatically optimized with the objective function of minimizing the validation error. In the last stage, the performance of the trained prediction model is validated for multi-step ahead prediction and a MC dropout technique is used to capture the prediction uncertainty. The procedure of the proposed framework can be summarized as in Algorithm 1, where *max_iter* is the number of optimization iterations and N_{MC} is the number of MC dropout realizations. The details of each stage are given in the following sections.

Algorithm 1. Procedure of the proposed multi-step prediction framework

---

Input: A raw time series data collected up to time t: $\{x_1, x_2, ..., x_t\}$

Output: Predictions of H next observations and their uncertainty information

Preprocessing stage

1. Detect and remove outliers
2. Impute missing data points

Model selection stage

3. **for** i in {1,…,*max_iter*} **do**
   a. Select the optimal network hyperparameters at the ith trial with TPE
   b. Validate the hyperparameters by using k-fold cross-validation
   c. Update the fitness value with the average training error measured over k folds
4. Select the best hyperparameter setting with the lowest fitness value

Multi-step ahead prediction stage

5. **for** i in {1,…,N_{MC}} **do**
   a. Build a LSTM-based prediction model with the selected hyperparameters
   b. Perform the predictions for H steps ahead $\{\hat{x}_{t+h}\}, h \in [1, H]$ by using the MIMO prediction strategy
6. Calculate the mean and confidence interval of the predictions over N_{MC} realizations

---

4.1 Data preprocessing

As mentioned in Section 1, the quality of the observation data for training is one of the most important factors for the successful performance of a prediction model. Due to the errors during sensor measurements or signal transmission, the acquired observations may include missing and anomalous data points, e.g. outliers, which can negatively impact the model performance. In this study, we adopt

a raw data preprocessing module focusing on the two following tasks: 1) detecting and removing outliers; 2) imputing missing data points, the number of which may increase after removing outliers.

The first problem is addressed by using the Isolation Forest, an outlier detection technique built on the basis of decision trees [66]. This technique is based on an assumption that outliers are few, different and susceptible to a mechanism called isolation. In comparison with conventional distance and density measures, isolation has been proved to be a much more effective indicator to detect anomalies. In addition, Isolation Forest also requires a small linear time complexity. Further details on the algorithm of Isolation Forest can be found in [66]. Once outliers are reduced, a local polynomial regression technique is used to reconstruct missing data samples and reduce noises. The preprocessed data is later used for training and testing the prediction model in the following stages.

4.2 Model selection

4.2.1 Prediction horizon

Several research works have been carried out on determining an optimal horizon of prediction in order to provide predictions accurately and timely, and to ensure the usefulness of the prognostic model. However, to the authors' knowledge, there is no general rule reported for dealing with this issue. We have carried out a review on the horizons selected in recent prediction studies for industrial applications during 2015-2019 [15], [58], [67]–[91] and the result is summarized in Fig. 5. The result shows that multi-step ahead prediction has been less studied than single-step ahead prediction, and that most of the works were carried out with horizons ranging from 3 to 6 steps ahead. To demonstrate the effectiveness of the proposed model, a prediction horizon of 15 steps ahead is investigated in this study.

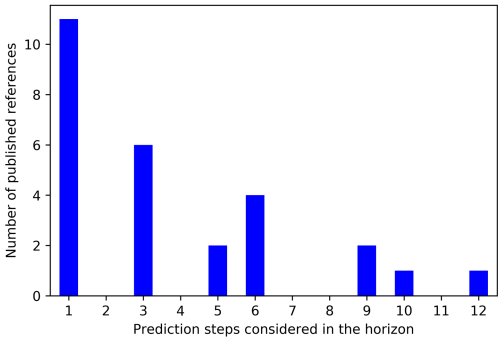


Fig. 5. Prediction horizons of recent studies.

4.2.2 Hyperparameter optimization

In machine learning, hyperparameters define the model architecture and control the learning process, e.g. the number of hidden layers, activation function type and learning rate. Automatic hyperparameter optimization is playing a fundamental role in the development of machine learning models, including the recent deep neural networks, e.g. LSTM, whose learning performance greatly depends on a number of hyperparameter choices [92]. Automatic hyperparameter optimization has several important advantages, such as: 1) reduction of the human effort in deploying machine learning, which is important in application because different hyperparameter configurations are needed for different datasets [93]; 2) improvement of the performance of machine learning models, by choosing the most appropriate (according to specified objectives) hyperparameters values for the target application at hand [94], [95]; 3) increase of the reproducibility of results, as automatic hyperparameter optimization is clearly more reproducible than manual tuning by human and allows fair comparisons between different models by giving them the same level of tuning for the specific application [96].

In this study, we implement a variant of Bayesian optimization (BO), called Tree-structured Parzen Estimator (TPE) [97], to automatically optimize the hyperparameters of the proposed prediction model.

A common advantage of BO approaches is that they require less function evaluations than other classical optimization approaches, such as grid search or RS. This is because these approaches learn and select the best hyperparameter sets based on their distributions describing the fitness scores in the previous iterations. Thus, the number of samples drawn from the hyperparameter search space is probabilistically guided and reduced, allowing for proper evaluations of the most promising candidates for hyperparameter choices.

Recently, TPE has been put forward to address the limitation of the conventional BO approaches in working with categorical and conditional parameters, and, thus, to improve the hyperparameters selection process [97]. It has, then, been widely used to tune machine learning models in various applications, such as image processing [96], [98]–[101], electricity price forecasting [102], solar irradiance forecasting [103], rail defect prediction [104], occupational accident prediction [105].

Parzen-window density estimation, which is also known as kernel density estimation, is a non-parametric way to build a probability density function from empirical data. In the TPE algorithm, each sample from the empirical data defines a Gaussian distribution with a mean equal to the hyperparameter value and a specified standard deviation. At the start-up iterations, a random search is employed to initialize the distributions by sampling the response surface $\{\theta^{(i)}, y^{(i)}, i = 1, \ldots, N_{init}\}$, where θ denotes the hyperparameter set, y is the corresponding value on the response surface, i.e. the validation loss or the fitness value, and N_{init} is the number of start-up iterations. Then, the hyperparameter space is divided into two groups, namely *good* and *bad* samples, based on their fitness values and a predefined threshold value y^* (usually set to 15% [92]), as follows:

$$p(\theta \mid y) = \begin{cases} \Pr_{good}(\theta) & \text{if } y < y^* \\ \Pr_{bad}(\theta) & \text{if } y \geq y^* \end{cases} \tag{18}$$

where $\Pr_{good}$ and $\Pr_{bad}$ are the probabilities that the hyperparameter set θ is in the *good* and *bad* groups, respectively. Fig. 6 illustrates an example of the TPE initialization process for the hyperparameter distributions, with $y^* = 15\%$ and $N_{\text{init}} = 100$. The red points are the samples with the lowest fitness values after evaluation, thus being classified into the *good* group $\Pr_{good}$ whereas the others form the *bad* group $\Pr_{bad}$. In this way, the selection of optimal hyperparameters does not rely on the best observation, but on a set of best observations and their distributions. Then, the more iterations one used for initialization, the better distribution we get at the beginning. An Expected Improvement (EI) is, then, calculated as follows:

$$EI(\theta) = \frac{\Pr_{good}(\theta)}{\Pr_{bad}(\theta)} \tag{19}$$

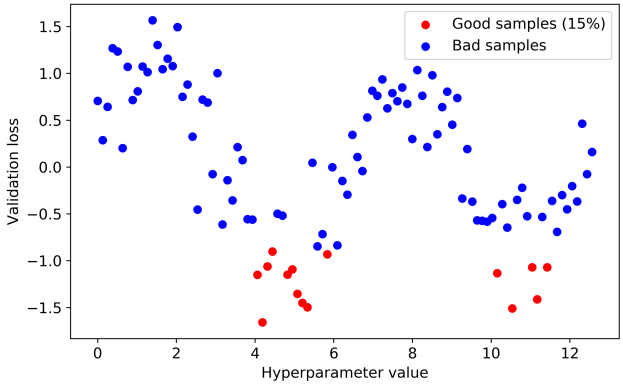


Fig. 6. Samples classification from the TPE initialization process.

At each iteration, the hyperparameter configuration $\theta*$ that maximizes the EI is chosen. Fig. 7 shows the flowchart of the TPE optimization procedure.

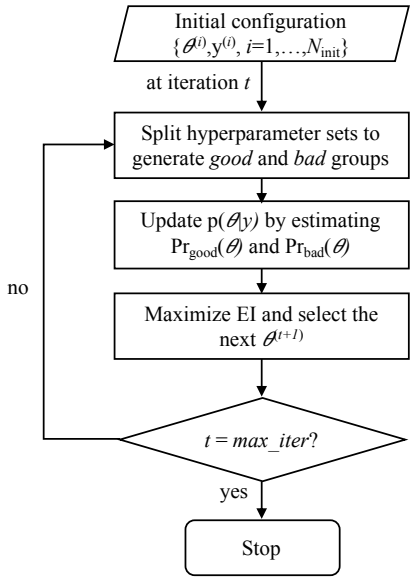


Fig. 7. Flowchart of the TPE optimization procedure.

4.3 Multi-step ahead prediction

In the testing stage, the MIMO prediction strategy introduced in Section 2.2.3, is used to predict the future values. As mentioned in Section 4.2.1, the prediction horizon h is set to 15-step ahead in this study, as shown in Fig. 8.

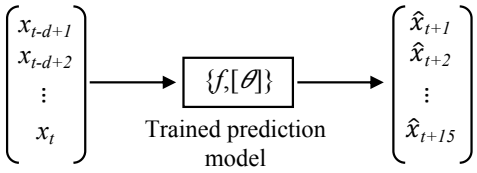


Fig. 8. Multi-step ahead prediction procedure.

To further assess the prediction performance, we adopt a Monte Carlo (MC) dropout technique [106] in order to capture the uncertainty information of the multi-step ahead predictions of the proposed model. It is important to note that the standard LSTM network is not capable to quantify the prediction uncertainty itself. In the MC dropout technique, a dropout probability is applied to all the weight layers in the network, which represents the network weights drawn from a Bernoulli distribution. Thus, the prediction uncertainty can be quantified by running several forward passes through the network. In this study, we perform N_{MC}=100 stochastic forward passes, in which network units of each layer are randomly dropped out, and obtain the mean and confidence interval of the predictions.

5 Experimental study

5.1 SG data

In this paper, the prediction performance of the proposed model is evaluated on the SG data of French NPPs. SGs in pressurized water reactors (PWRs) are heat exchangers which use the heat from the primary reactor coolant to produce steam in the secondary side and, thus, drive the turbine generators. In addition, the SGs act as a safety barrier between the radioactive primary side and the non-radioactive secondary side. Due to their critical role in NPPs, any degradation mechanism in SGs should be monitored and prevented at the early stages of propagation. A widely used method of degradation monitoring is the analysis of the wide range level (WRL) dynamic behavior recorded by control sensors [107], [108].

WRL is one of the condition monitoring variables measured from the NPP SGs. It is estimated from the difference between the pressure measured at two difference heights, i.e. the dome and the bottom of the downcomer, as illustrated in Fig. 9 (label 18) [108]. Due to its nature, WRL is very sensitive to

the temperature, the flow rate of the feed-water and the circulation ratio of the SG. Usually, WRL variations are monitored during slow transients and during manual control at low power load [108]. Among critical SG degradation mechanisms, clogging is a phenomenon where the flow holes of the tube support plates are partially or completely blocked by deposits, leading to the reduction of the circulation flow rate in the SGs [108]. Clogging in SG is a slow process which may take several years. In [109], it has been shown that the WRL of a SG is closely related to the clogging degradation. Thus, the predictions of WRL can be converted to the clogging degradation state.

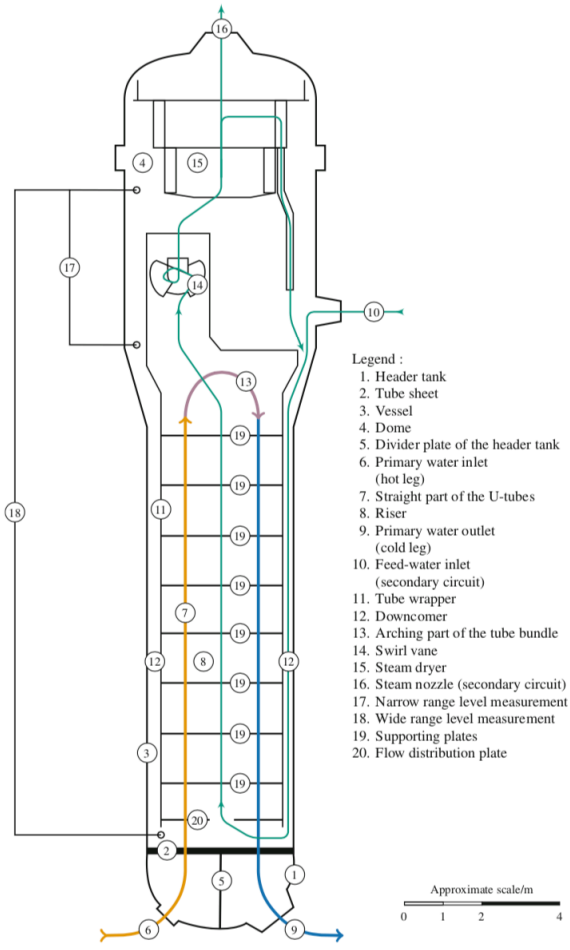
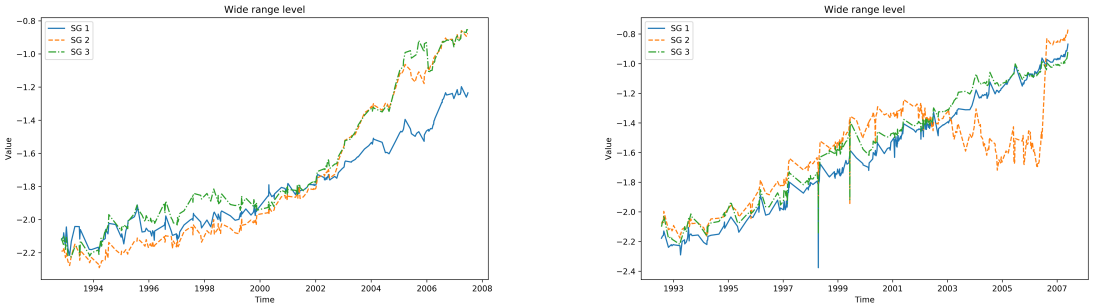


Fig. 9. The front-cut schematic of a 51B-model SG [108].

The original SG data employed in this study were collected from six SGs of two different 900-MW NPPs, which are operated by Électricité de France (EDF). Each plant consists of three SGs. The WRL data were recorded during the stationary regimes in which the power demand percentage is stably maintained greater than 90%, at an interval of 3 days from July 1992 to June 2007. Fig. 10 shows the

temporal evolution of the WRL observations of the two NPPs. The names of the plants are omitted for confidentiality reasons.

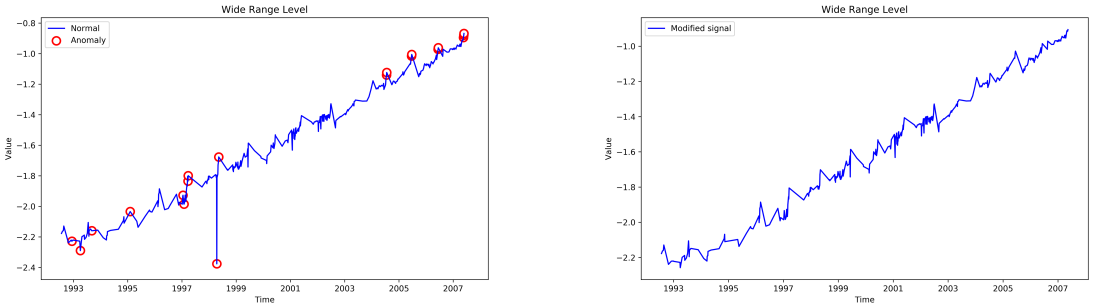


(a) Plant No. 1 (b) Plant No. 2

Fig. 10. Raw WRL measurements recorded from control sensors of different NPPs.

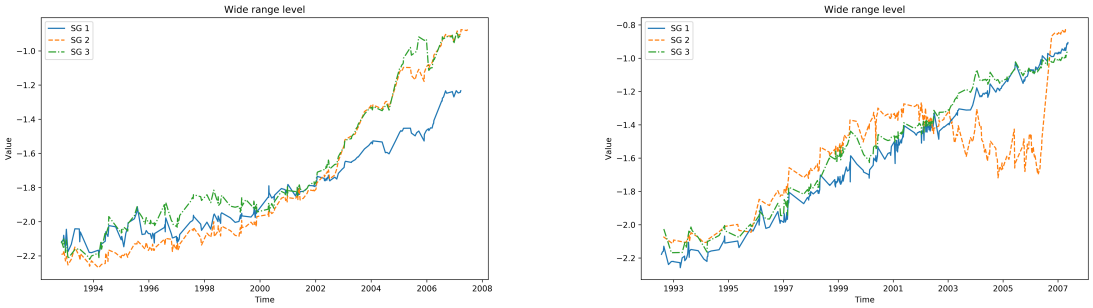
5.2 Data preprocessing

Before being used for the model development, the raw SG data are preprocessed by using the Isolation Forest and local regression approaches described in Section 4.1. Fig. 11 shows the results of applying the Isolation Forest for reducing outliers in the data of SG 1 of plant No. 2. In Fig. 11(a), the solid line indicates the normal measurements whereas the detected outliers are highlighted as circled points, which are later eliminated in Fig. 11(b). An interesting observation in Fig. 11(a) is the anomalous spike between 1997 and 1999. Without the outlier detection step, this sudden spike could highly impact, in a negative manner, on the prediction accuracy. After reducing the outliers, imputations for missing data samples are given. The preprocessed data of all SGs after the preprocessing stage are shown in Fig. 12.



(a) Original signal with outliers highlighted (b) Modified signal after outlier removal

Fig. 11. Applying the Isolation Forest to the data of SG 1 of plant No. 2.



(a) Plant No. 1 (b) Plant No. 2

Fig. 12. The results of the preprocessing stage for all SG data.

6 Results and discussion

After the preprocessing stage, each SG data series is divided into a training set and a testing set. The data for the first 11 years, from July 1992 to December 2002, which include a total of 1230 samples at a 3-day interval, are selected to train the proposed prediction model and the next 5-year data with 510 samples are employed to test the model performance.

Before constructing the proposed model, we employ the false nearest neighbor (FNN) algorithm [110] to determine the appropriate embedding dimension d of the data series. The main idea of the FNN algorithm is to find the minimum dimension where the distances between the nearest neighbors in the time series do not significantly change in the next higher dimensional embedding. Fig. 13 shows the result of applying FNN to the data of SG 1 of plant No. 1. A threshold for identifying the minimum embedding dimension is set to 0. In this Figure, the minimum embedding dimension value is found at 12. We summarize the optimal embedding dimensions identified for all the SGs data series in Table 1.

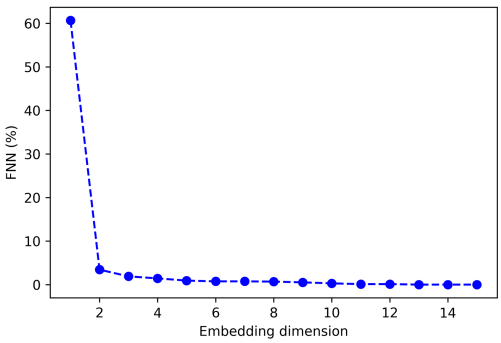


Fig. 13. FNN result for SG 1 of plant No. 1.

Table 1. Minimum embedding dimensions for all SGs.

Plant	No. 1			No. 2		
SG	1	2	3	1	2	3
Embedding dimension	12	13	9	9	11	6

In this study, we carry out three comparisons to evaluate the performance of the proposed prognostic model. The first comparison is conducted to analyze the viability of TPE in tuning the proposed model during the training stage. As a standard optimization approach, RS is considered for benchmarking purposes. Another comparison is, then, carried out to specifically validate the efficacy of dropout in the proposed prediction framework. In the third comparison, four hybrid prediction models, including single-output support vector regression using recursive strategy (SVR-REC), multi-output support vector regression using MIMO strategy (SVR-MIMO), single-output multilayer perceptron neural network using recursive strategy (MLP-REC) and multi-output multilayer perceptron neural network using MIMO strategy (MLP-MIMO), are employed as the benchmark models for comparison with the proposed model in multi-step ahead predictions. In this performance evaluation, three prediction accuracy metrics are considered, including root mean square error (RMSE), mean absolute percentage error (MAPE) and mean absolute scaled error (MASE). Their definitions are given as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{x}_i - x_i\right)^2}, \tag{20}$$

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{\hat{x}_i - x_i}{x_i}\right| \times 100\%, \tag{21}$$

$$MASE = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\left|\hat{x}_i - x_i\right|}{\frac{1}{N-1}\sum_{j=2}^{N}\left|x_j - x_i\right|}\right), \tag{22}$$

where N is the number of testing observations, x and $\hat{x}$ are the observed and predicted values, respectively.

6.1 Automatic hyperparameter optimization

The proposed prediction model is constructed with one LSTM layer with 64 neurons. Four major hyperparameters of the model are to be tuned, including dropout rate, activation function type, optimizer type and learning rate. The details of the hyperparameter search space are shown in Table 2. For a fair comparison, the TPE and RS algorithms are evaluated by using the same model configurations and hyperparameter search space. The number of optimization trials is selected as 30 for the two algorithms. In addition, a *k*-fold cross-validation ($k = 3$ in this study) is adopted to prevent overfitting during training the model. The mean square error (MSE) is used as the objective function for model selection. In other words, at each optimization trial, the hyperparameter configuration with the lowest average prediction error evaluated by cross-validation is chosen. To achieve the training convergence, the number of training epochs is set to 100.

Table 2. Hyperparameters of the proposed prediction model.

Hyperparameter	Type of distribution	Value set or Range
Dropout rate	Uniform float	[0, 0.5]
Activation function	Categorical	{Linear, Sigmoid, Tanh, ReLU}
Optimizer	Categorical	{SGD, RMSprop, Adam}
Learning rate	Uniform float	[0.0001, 0.1]

Fig. 14 shows the comparison of the TPE and RS hyperparameter searches over 30 trials for SG 1 of plant No. 1. The corresponding training loss is also given in Fig. 15. In particular, the TPE algorithm uses the first 20 startup trials for initializing the distributions of the *good* and *bad* hyperparameter sets, as mentioned in Section 4.2.2. This initialization process is performed by employing a standard RS. Therefore, in Figs. 14 and 15, we can observe a similar performance between TPE and RS in both hyperparameter searching and their obtained training losses during the first 20 trials. However, the performance of TPE is quickly improved after the initialization. It much more focuses on the good hyperparameter configurations which was found in the previous trials, leading to faster converge and lower training loss than RS within 30 trials.

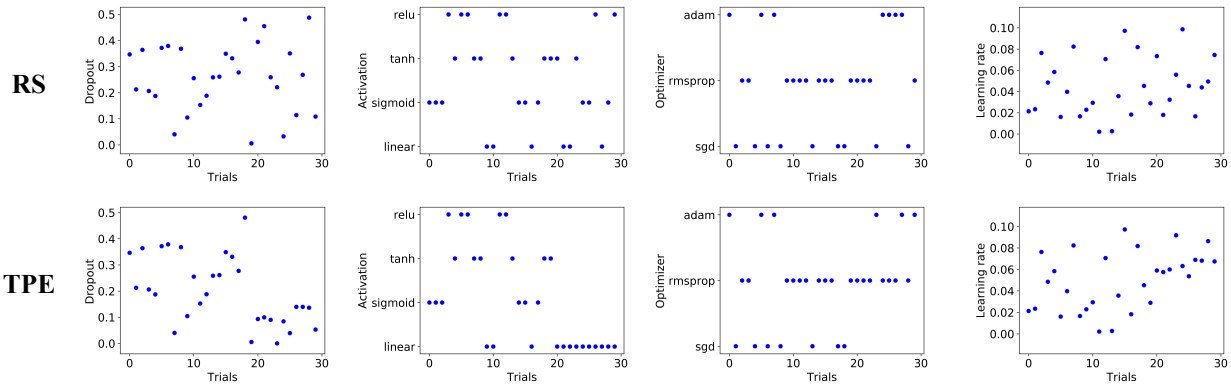


Fig. 14. Hyperparameters tuning process over 30 trials by TPE (top Figures) and RS (bottom Figures) for SG 1 of plant No.1.

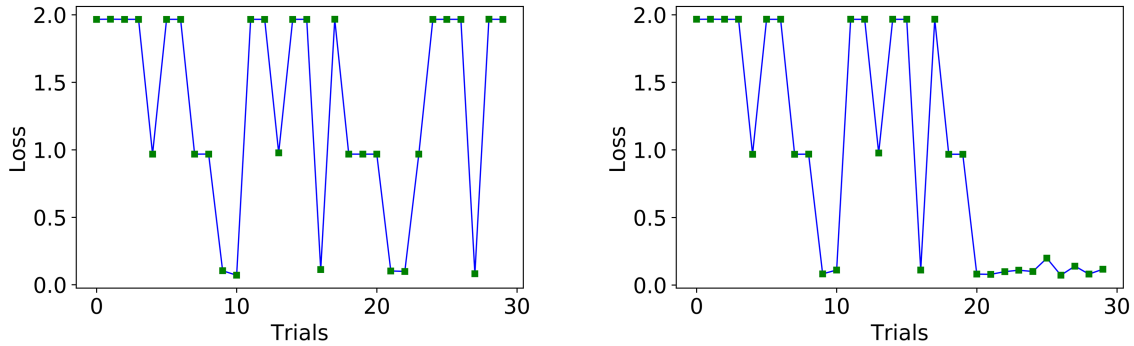


Fig. 15. Training loss versus trials of TPE and RS for SG 1 of plant No. 1.

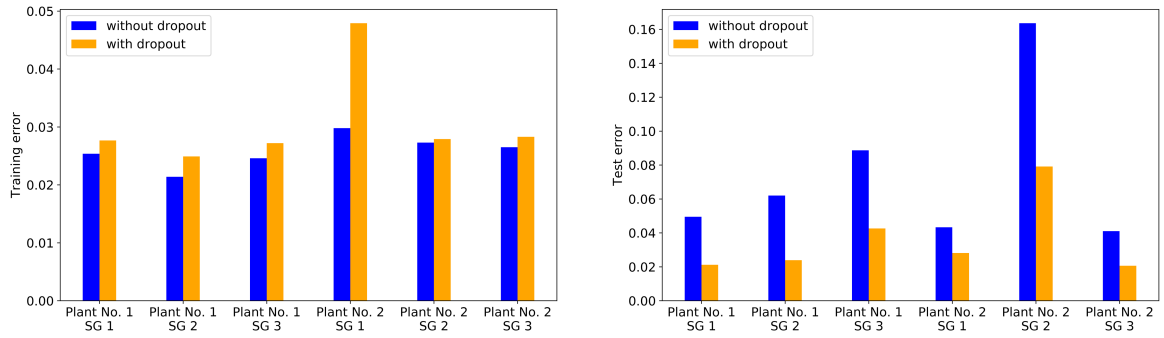
In Table 3, we show the performance comparison between TPE and RS, in terms of their obtained best training loss for all SGs. The results obviously show that the optimal configurations found by TPE generally outperform the best ones found by RS in the considered case studies. Thus, the optimal hyperparameter configurations found by TPE are used for prediction in the next stage.

Table 3. The best training loss obtained by TPE and RS in hyperparameter tuning for all SGs.

Plant	No. 1			No. 2		
SG	1	2	3	1	2	3
Random search	0.0487	0.0479	0.0307	0.0358	0.0321	0.0343
TPE	0.0440	0.0370	0.0319	0.0350	0.0314	0.0270

6.2 Dropout regularization

In this section, a comparison is carried out between the proposed prediction model and a model with the same architecture but trained without dropout. The other hyperparameters are kept identical between the two models, as described in Section 6.1. The probability of the used dropout is automatically optimized by TPE. We employ all the six SG datasets to comprehensively evaluate dropout during both the training and test phases in terms of RMSE. The comparative results are shown in Fig. 16. The result shows that the prediction model trained without dropout has lower training errors but much higher test errors, which may be an indication of the presence of overfitting. In contrast, the dropout model significantly reduces the overfitting problem with lower test errors for all the datasets. The average error reduction of the dropout model is 51.91%, which strongly indicates the efficacy of dropout in reducing overfitting and improving the prediction performance of the neural network.



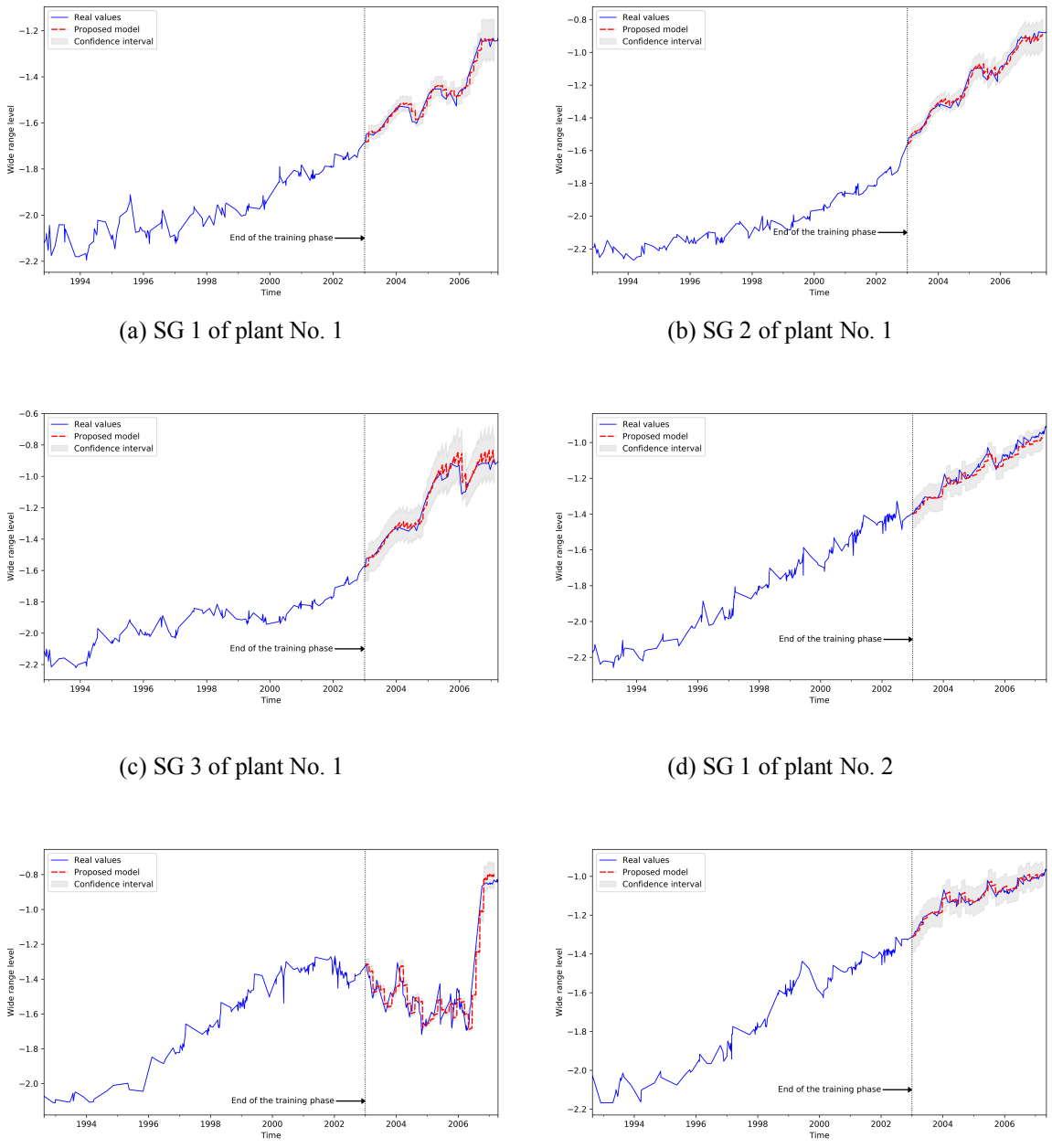
(a) Training phase (b) Test phase

Fig. 16. Training and test errors for the network architecture trained without and with dropout.

6.3 Performance evaluation

The WRL measurements of the six SGs are used for validating the developed prediction model for multi-step ahead prediction. It is important to remind that the prediction horizon used in this study is 15 steps ahead, which equals 45 operating days of the SGs. After the training is finished, the prediction model is used to continuously predict 15-step ahead in the next 5 years. Fig. 17 illustrates the prediction results of the proposed model for all SGs. The predicted values are shown as the dashed line, whereas the solid line depicts the actual observations. The 95% confidence interval of the predictions, obtained via MC simulations, is depicted as the grey region. The results show that the proposed model is able to

keep track with the changes of the WRL data while achieving accurate predictions, which are very close to the actual data for all SGs. Moreover, the 95% confidence bounds of the predictions are narrow and close to the target values, indicating predictions with a high precision. In industrial applications, these results are of crucial importance for accurately estimating the equipment RUL.



(a) SG 1 of plant No. 1

(b) SG 2 of plant No. 1

(c) SG 3 of plant No. 1

(d) SG 1 of plant No. 2

(e) SG 2 of plant No. 2

(f) SG 3 of plant No. 2

Fig. 17. Multi-step ahead prediction results by the proposed model for all SGs.

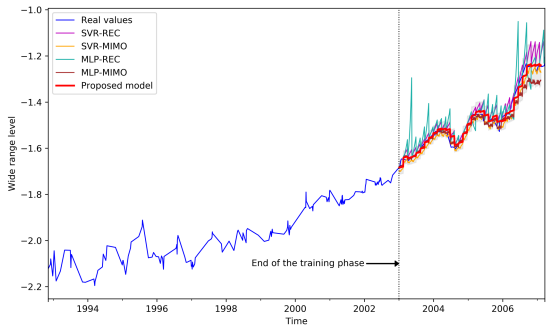
The prediction results obtained by the proposed model are, then, evaluated with respect to the four

benchmark models, i.e. SVR-REC, SVR-MIMO, MLP-REC and MLP-MIMO, in terms of prediction accuracy. For a fair comparison, the hyperparameters of the compared models are optimized by using TPE with 30 trials. The details of the hyperparameter search spaces of the compared models are shown in Table 4.

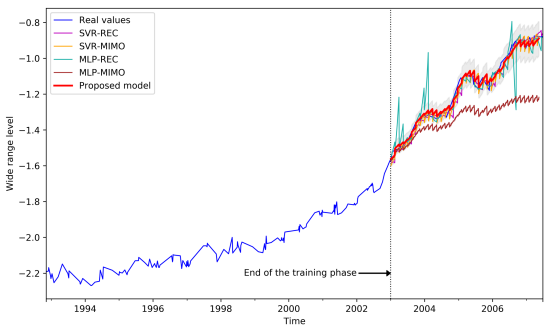
Table 4. Hyperparameters of the benchmark models.

Model	Hyperparameter	Value set or Range
SVR (including SVR-REC and SVR-MIMO)	Kernel function	{Linear, RBF, Poly, Sigmoid}
	Degree (of the polynomial kernel function)	[2, 4]
	Regularization parameter (C)	[0.01, 100]
	Kernel coefficient (gamma)	[0.01, 10]
MLP (including MLP-REC and MLP-MIMO)	Hidden layer size	[1, 5]
	Activation function	{Logistic, Tanh, ReLU}
	Optimizer	{LBFGS, SGD, Adam}
	Learning rate	{Constant, Invscaling, Adaptive}
	Regularization parameter (alpha)	[0.0001, 0.01]

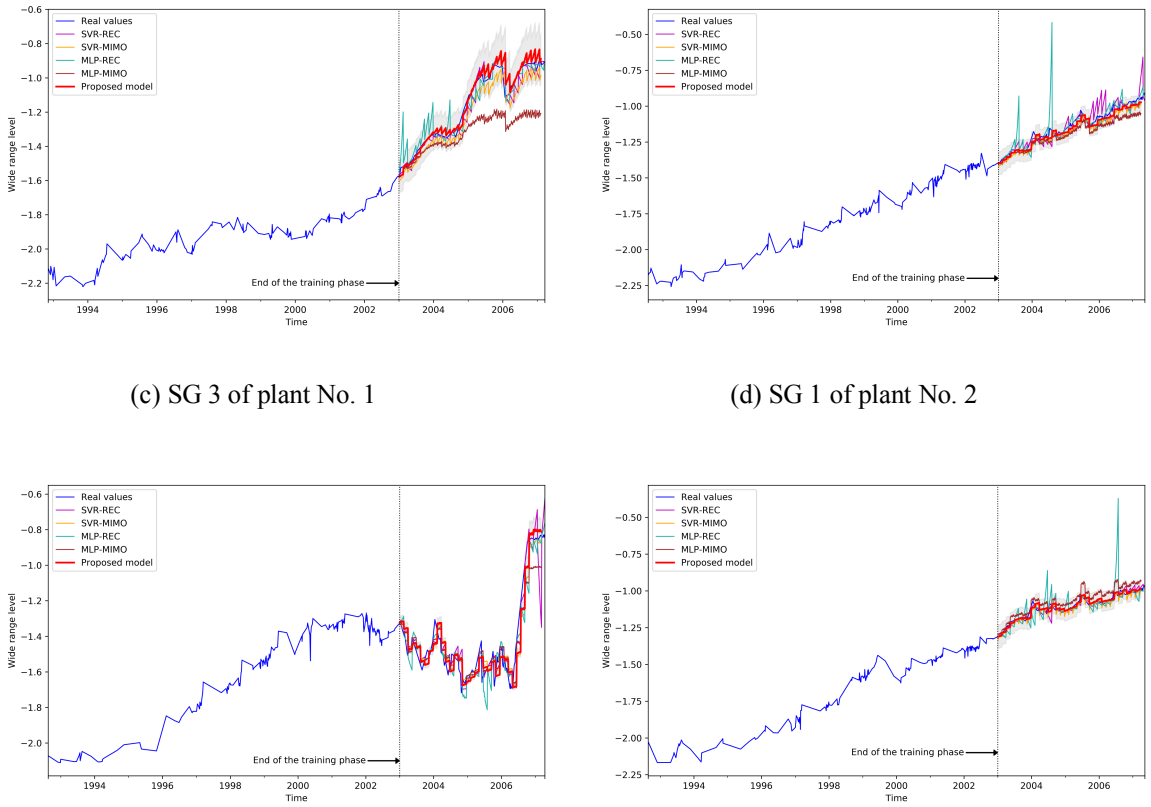
The comparative results of the proposed model and the four benchmark models for multi-step ahead predictions are shown in Fig. 18. Table 5 summarizes the prediction results in terms of the three accuracy indicators for different SG data. As can be seen in Fig. 18 and Table 5 (values in bold), the proposed prediction model outperforms the four other benchmark models and achieves higher accuracy for all SGs. The results indicate the accurate and efficient learning of the proposed prediction model for the long-term dependencies of the SG data.



(a) SG 1 of plant No. 1



(b) SG 2 of plant No. 1



(c) SG 3 of plant No. 1

(d) SG 1 of plant No. 2

(e) SG 2 of plant No. 2

(f) SG 3 of plant No. 2

Fig. 18. Multi-step ahead predictions using different models for all SGs.

Table 5. Comparison of the prediction performance in multi-step ahead predictions for all SGs.

	Method	SG 1			SG 2			SG 3		
		RMSE	MAPE	MASE	RMSE	MAPE	MASE	RMSE	MAPE	MASE
Plant No. 1	SVR-REC	0.0382	2.0775	15.2484	0.0333	2.1970	10.2221	0.0508	3.3173	12.1521
	SVR-MIMO	0.0283	1.6511	12.2085	0.0331	2.3878	10.8281	0.0640	5.1793	19.2863
	MLP-REC	0.0597	2.7824	20.8682	0.0656	3.3349	15.3398	0.0577	3.3597	13.4886
	MLP-MIMO	0.0339	1.7074	11.8887	0.1888	15.1554	62.2885	0.1867	15.2453	52.7662
	Proposed model	**0.0212**	**1.0950**	**8.6166**	**0.0239**	**1.6973**	**5.8214**	**0.0426**	**2.7230**	**4.0846**
Plant No. 2	SVR-REC	0.0572	3.6462	12.8555	0.0906	4.7909	**7.0354**	0.0242	1.5005	**6.1132**
	SVR-MIMO	0.0401	3.1774	11.8732	0.0849	4.7575	7.4570	0.0247	1.7842	7.5984
	MLP-REC	0.0751	3.0309	11.2367	0.0862	4.7819	7.4403	0.0734	3.4535	14.2511
	MLP-MIMO	0.0607	4.8530	17.4741	0.0888	5.4251	7.5992	0.0499	4.1168	17.2689
	Proposed model	**0.0281**	**2.0117**	**8.6455**	**0.0791**	**4.4033**	9.3923	**0.0206**	**1.3992**	7.9604

The average computational time of training the proposed prediction model is 3.2 hours, on a GPGPU node comprising 2 Intel Xeon CPU E5-2695 (24 cores at 2.40 Hz with 32 GB of RAM) and 2

Nvidia Tesla K40m graphic cards (with 12 GB of GRAM). It is important to note that SG data used in this paper were recorded at an interval of 3 days. After being trained, the proposed model can be used to perform a 15-step ahead prediction, which is equivalent to 45 operating days ahead of the SGs. Due to this reason, the proposed prediction framework can be applied for a real-time time series prediction of the considered application.

The authors have tested the proposed framework on data from several nuclear power plants, with satisfactory results. Unfortunately, for industrial confidentiality, the data cannot be disclosed and shared.

7 Conclusion and Future Work

This paper presents an original multi-step ahead prediction framework for PHM applications. The framework integrates three consecutive steps: (1) data preprocessing, (2) adaptive model building and (3) multi-step ahead prediction. Initially, the problems of abnormal outliers and missing data samples are addressed by employing two preprocessing techniques: Isolation Forest and local regression. Then, a LSTM RNN is constructed for making predictions over a long-term horizon, in which the network hyperparameters are automatically optimized by a TPE algorithm. A dropout regularization and a cross-validation techniques are applied to address the overfitting problem during the training phase. Finally, the performance of the proposed model is evaluated for multi-step ahead predictions with a MIMO prediction strategy employed. A MC dropout is adopted to quantify the prediction uncertainty.

The proposed multi-step ahead prediction framework can be used for the predictions of time series of NPP operating parameters. A case study concerning the real WRL measurements of SGs which were acquired from different NPPs in France over a period of 16 years is carried out for validating the proposed framework. The experimental results show that the developed prediction framework is able to adaptively estimate the optimal setting for its architecture and capture the underlying long-term dependencies inherent in the given data, for achieving accurate predictions over a long horizon, up to 45 days ahead, outperforming conventional prediction approaches.

However, for the application of NPP SGs used in this study, sufficient information and data for

performing a multivariate prediction are not provided, e.g. the information of the interdependency between measured variables and degradations (or failures), the interdependency within the variables, and the maintenance reports of the NPP SGs. Future research will be performed to develop a multivariate time series prediction model and integrate the proposed framework within a RUL estimation task for PHM and predictive maintenance.

Acknowledgement

The authors would like to thank the PRISME department of Électricité de France (EDF) R&D for providing the data used in this paper.

References

[1] E. Zio, "Prognostics and Health Management of Industrial Equipment," in *Diagnostics and Prognostics of Engineering Systems: Methods and Techniques*, S. Kadry, Ed. IGI Global, 2012, pp. 333–356.

[2] R. Gouriveau and N. Zerhouni, "Connexionist-systems-based long term prediction approaches for prognostics," *IEEE Trans. Reliab.*, 2012.

[3] A. Sorjamaa, J. Hao, N. Reyhani, Y. Ji, and A. Lendasse, "Methodology for long-term prediction of time series," *Neurocomputing*, vol. 70, no. 16–18, pp. 2861–2869, Oct. 2007.

[4] A. S. Weigend, *Time Series Prediction: Forecasting The Future And Understanding The Past*. New York: Routledge, 1994.

[5] J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, and D. Siegel, "Prognostics and health management design for rotary machinery systems - Reviews, methodology and applications," *Mech. Syst. Signal Process.*, 2014.

[6] H. P. Nguyen, J. Liu, and E. Zio, "Ensemble of Models for Fatigue Crack Growth Prognostics," *IEEE Access*, vol. 7, pp. 49527–49537, 2019.

[7] H. P. Nguyen, J. Liu, and E. Zio, "Dynamic-weighted ensemble for fatigue crack degradation state prediction," *Eng. Fract. Mech.*, 2018.

[8] Y. Hu, P. Baraldi, F. Di Maio, and E. Zio, "A particle filtering and kernel smoothing-based approach for new design component prognostics," *Reliab. Eng. Syst. Saf.*, vol. 134, pp. 19–31, 2014.

[9] L. Liao and F. Köttig, "A hybrid framework combining data-driven and model-based methods for system remaining useful life prediction," *Appl. Soft Comput. J.*, 2016.

[10] H. Nguyen, W. Fauriat, E. Zio, and J. Liu, "A Data-Driven Approach for Predicting the Remaining Useful Life of Steam Generators," in *2018 3rd International Conference on System Reliability and Safety (ICSRS)*, 2018, pp. 255–260.

[11] Y. Xiao and L. Feng, "A novel neural-network approach of analog fault diagnosis based on kernel discriminant analysis and particle swarm optimization," *Appl. Soft Comput. J.*, 2012.

[12] Y. Lin, X. Li, and Y. Hu, "Deep diagnostics and prognostics: An integrated hierarchical learning framework in PHM applications," *Appl. Soft Comput. J.*, 2018.

[13] J. Yu, "A hybrid feature selection scheme and self-organizing map model for machine health assessment," *Appl. Soft Comput. J.*, 2011.

[14] J. Sanz, R. Perera, and C. Huerta, "Gear dynamics monitoring using discrete wavelet transformation and multi-layer perceptron neural networks," *Appl. Soft Comput. J.*, 2012.

[15] J. Liu, V. Vitelli, E. Zio, and R. Seraoui, "A Novel Dynamic-Weighted Probabilistic Support Vector Regression-Based Ensemble for Prognostics of Time Series Data," *IEEE Trans. Reliab.*, 2015.

[16] A. Rai and S. H. Upadhyay, "An integrated approach to bearing prognostics based on EEMD-multi feature extraction, Gaussian mixture models and Jensen-Rényi divergence," *Appl. Soft Comput. J.*, 2018.

[17] L. L. Li, Z. F. Liu, M. L. Tseng, and A. S. F. Chiu, "Enhancing the Lithium-ion battery life predictability using a hybrid method," *Appl. Soft Comput. J.*, 2019.

[18] F. O. Heimes, "Recurrent neural networks for remaining useful life estimation," in *2008 International Conference on Prognostics and Health Management, PHM 2008*, 2008.

[19] P. J. Angeline, G. M. Saunders, and J. B. Pollack, "An Evolutionary Algorithm that Constructs Recurrent Neural Networks," *IEEE Trans. Neural Networks*, 1994.

[20] Z. Li, D. Wu, C. Hu, and J. Terpenny, "An ensemble learning-based prognostic approach with degradation-dependent weights for remaining useful life prediction," *Reliab. Eng. Syst. Saf.*, 2019.

[21] J. Chen, H. Jing, Y. Chang, and Q. Liu, "Gated recurrent unit based recurrent neural network for remaining useful life prediction of nonlinear deterioration process," *Reliab. Eng. Syst. Saf.*, 2019.

[22] E. Zio, M. Broggi, L. R. Golea, and N. Pedroni, "Failure and reliability predictions by infinite impulse response locally recurrent neural networks," *Chem. Eng. Trans.*, 2012.

[23] A. E. R. ElSaid, F. El Jamiy, J. Higgins, B. Wild, and T. Desell, "Optimizing long short-term memory recurrent neural networks using ant colony optimization to predict turbine engine vibration," *Appl. Soft Comput. J.*, 2018.

[24] Y. Zhang, R. Xiong, H. He, and M. G. Pecht, "Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries," *IEEE Trans. Veh. Technol.*, 2018.

[25] J. Liu, A. Saxena, K. Goebel, B. Saha, and W. Wang, "An Adaptive Recurrent Neural Network for Remaining Useful Life Prediction of Lithium-ion Batteries," *Natl. Aeronaut. Sp. Adm. Moffett F. Ca Ames Res. Cent.*, 2010.

[26] D. Liu, W. Xie, H. Liao, and Y. Peng, "An integrated probabilistic approach to lithium-ion battery

remaining useful life estimation," *IEEE Trans. Instrum. Meas.*, 2015.

[27] Y. Cheng, H. Zhu, J. Wu, and X. Shao, "Machine Health Monitoring Using Adaptive Kernel Spectral Clustering and Deep Long Short-Term Memory Recurrent Neural Networks," *IEEE Trans. Ind. Informatics*, 2019.

[28] B. Zhang, S. Zhang, and W. Li, "Bearing performance degradation assessment using long short-term memory recurrent network," *Comput. Ind.*, 2019.

[29] L. Ren, X. Cheng, X. Wang, J. Cui, and L. Zhang, "Multi-scale Dense Gate Recurrent Unit Networks for bearing remaining useful life prediction," *Futur. Gener. Comput. Syst.*, 2019.

[30] A. Malhi, R. Yan, and R. X. Gao, "Prognosis of defect propagation based on recurrent neural networks," *IEEE Trans. Instrum. Meas.*, 2011.

[31] R. Ma, T. Yang, E. Breaz, Z. Li, P. Briois, and F. Gao, "Data-driven proton exchange membrane fuel cell degradation predication through deep learning method," *Appl. Energy*, 2018.

[32] K. Javed, R. Gouriveau, N. Zerhouni, and D. Hissel, "Prognostics of Proton Exchange Membrane Fuel Cells stack using an ensemble of constraints based connectionist networks," *J. Power Sources*, 2016.

[33] L. Guo, N. Li, F. Jia, Y. Lei, and J. Lin, "A recurrent neural network based health indicator for remaining useful life prediction of bearings," *Neurocomputing*, vol. 240, pp. 98–109, May 2017.

[34] N. A. Gershenfeld, A. S. Weigend, N. A. Gershenfeld, and A. S. Weigend, "The future of time series," *Time Ser. Predict. Forecast. Futur. Underst. Past*, 1993.

[35] G. U. Yule, "On a Method of Investigating Periodicities in Disturbed Series, with Special Reference to Wolfer's Sunspot Numbers," *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.*, 1927.

[36] E. Erdem and J. Shi, "ARMA based approaches for forecasting the tuple of wind speed and direction," *Appl. Energy*, 2011.

[37] B. Doucoure, K. Agbossou, and A. Cardenas, "Time series prediction using artificial wavelet neural network and multi-resolution analysis: Application to wind speed data," *Renew. Energy*, 2016.

[38] R. G. Kavasseri and K. Seetharaman, "Day-ahead wind speed forecasting using f-ARIMA models," *Renew. Energy*, 2009.

[39] C. D. Zuluaga, M. A. Álvarez, and E. Giraldo, "Short-term wind speed prediction based on robust Kalman filtering: An experimental comparison," *Appl. Energy*, 2015.

[40] M. Wu, C. Stefanakos, Z. Gao, and S. Haver, "Prediction of short-term wind and wave conditions for marine operations using a multi-step-ahead decomposition-ANFIS model and quantification of its uncertainty," *Ocean Eng.*, 2019.

[41] I. Kaushik and S. Singh, "Seasonal ARIMA Model for Forecasting of Monthly Rainfall and Temperature," *J. Environ. Res. Dev.*, 2008.

[42] M. A. Mariño, J. C. Tracy, and S. A. Taghavi, "Forecasting of reference crop evapotranspiration," *Agric. Water Manag.*, 1993.

[43] M. Ohyver and H. Pudjihastuti, "Arima Model for Forecasting the Price of Medium Quality Rice to

Anticipate Price Fluctuations," in *Procedia Computer Science*, 2018.

[44] Q. Wang, X. Song, and R. Li, "A novel hybridization of nonlinear grey model and linear ARIMA residual correction for forecasting U.S. shale oil production," *Energy*, 2018.

[45] M. Qin, Z. Li, and Z. Du, "Red tide time series forecasting by combining ARIMA and deep belief network," *Knowledge-Based Syst.*, 2017.

[46] J. Wang and Y. Li, "Multi-step ahead wind speed prediction based on optimal feature extraction, long short term memory neural network and error correction strategy," *Appl. Energy*, 2018.

[47] J. Berbić, E. Ocvirk, D. Carević, and G. Lončar, "Application of neural networks and support vector machine for significant wave height prediction," *Oceanologia*, 2017.

[48] B. Kamranzad, A. Etemad-Shahidi, and M. H. Kazeminezhad, "Wave height forecasting in Dayyer, the Persian Gulf," *Ocean Eng.*, 2011.

[49] J. Wang and Y. Li, "Short-Term Wind Speed Prediction Using Signal Preprocessing Technique and Evolutionary Support Vector Regression," *Neural Process. Lett.*, 2018.

[50] G. W. Chang, H. J. Lu, Y. R. Chang, and Y. D. Lee, "An improved neural network-based approach for short-term wind speed and power forecast," *Renew. Energy*, 2017.

[51] P. Jain and M. C. Deo, "Real-time wave forecasts off the western Indian coast," *Appl. Ocean Res.*, 2007.

[52] J. Wang and Y. Li, "An innovative hybrid approach for multi-step ahead wind speed prediction," *Appl. Soft Comput. J.*, 2019.

[53] G. Li and J. Shi, "On comparing three artificial neural networks for wind speed forecasting," *Appl. Energy*, 2010.

[54] J. P. S. Catalão, H. M. I. Pousinho, and V. M. F. Mendes, "Hybrid wavelet-PSO-ANFIS approach for short-term wind power forecasting in Portugal," *IEEE Trans. Sustain. Energy*, 2011.

[55] M. Özger and Z. Şen, "Prediction of wave parameters by using fuzzy logic approach," *Ocean Eng.*, 2007.

[56] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi, "Time series forecasting using a deep belief network with restricted Boltzmann machines," *Neurocomputing*, 2014.

[57] H. zhi Wang, G. qiang Li, G. bing Wang, J. chun Peng, H. Jiang, and Y. tao Liu, "Deep learning based ensemble approach for probabilistic wind power forecasting," *Appl. Energy*, 2017.

[58] Y. Li, H. Wu, and H. Liu, "Multi-step wind speed forecasting using EWT decomposition, LSTM principal computing, RELM subordinate computing and IEWT reconstruction," *Energy Convers. Manag.*, 2018.

[59] D. R. Cox, "Prediction by Exponentially Weighted Moving Averages and Related Methods," *J. R. Stat. Soc. Ser. B*, 1961.

[60] S. Ben Taieb, G. Bontempi, A. Sorjamaa, and A. Lendasse, "Long-term prediction of time series by combining direct and MIMO strategies," in *Proceedings of the International Joint Conference on Neural Networks*, 2009.

[61] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, 1997.

[62] C. Olah, "Understanding LSTM Networks," 2015. [Online]. Available:

https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[63] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent Neural Network Regularization," *arXiv e-prints*, p. arXiv:1409.2329, Sep. 2014.

[64] I. Sutskever, G. Hinton, A. Krizhevsky, and R. R. Salakhutdinov, "Dropout : A Simple Way to Prevent Neural Networks from Overfitting," *J. Mach. Learn. Res.*, 2014.

[65] A. Livnat, C. Papadimitriou, N. Pippenger, and M. W. Feldman, "Sex, mixability, and modularity," *Proc. Natl. Acad. Sci. U. S. A.*, 2010.

[66] F. T. Liu, K. M. Ting, and Z. H. Zhou, "Isolation forest," in *Proceedings - IEEE International Conference on Data Mining, ICDM*, 2008.

[67] W. Zhang, Z. Qu, K. Zhang, W. Mao, Y. Ma, and X. Fan, "A combined model based on CEEMDAN and modified flower pollination algorithm for wind speed forecasting," *Energy Convers. Manag.*, 2017.

[68] S. R. Moreno and L. dos Santos Coelho, "Wind speed forecasting approach based on Singular Spectrum Analysis and Adaptive Neuro Fuzzy Inference System," *Renew. Energy*, 2018.

[69] X. Mi, H. Liu, and Y. Li, "Wind speed prediction model using singular spectrum analysis, empirical mode decomposition and convolutional support vector machine," *Energy Convers. Manag.*, 2019.

[70] T. Niu, J. Wang, K. Zhang, and P. Du, "Multi-step-ahead wind speed forecasting based on optimal feature selection and a modified bat algorithm with the cognition strategy," *Renew. Energy*, 2018.

[71] J. Chen, G. Q. Zeng, W. Zhou, W. Du, and K. Di Lu, "Wind speed forecasting using nonlinear-learning ensemble of deep learning time series prediction and extremal optimization," *Energy Convers. Manag.*, 2018.

[72] C. Tian, Y. Hao, and J. Hu, "A novel wind speed forecasting system based on hybrid data preprocessing and multi-objective optimization," *Appl. Energy*, 2018.

[73] Z. Yang and J. Wang, "A hybrid forecasting approach applied in wind speed forecasting based on a data processing strategy and an optimized artificial intelligence algorithm," *Energy*, vol. 160, pp. 87–100, Oct. 2018.

[74] M. Santhosh, C. Venkaiah, and D. M. Vinod Kumar, "Ensemble empirical mode decomposition based adaptive wavelet neural network method for wind speed prediction," *Energy Convers. Manag.*, 2018.

[75] J. Naik, P. Satapathy, and P. K. Dash, "Short-term wind speed and wind power prediction using hybrid empirical mode decomposition and kernel ridge regression," *Appl. Soft Comput. J.*, 2018.

[76] Z. Yang and J. Wang, "A combination forecasting approach applied in multistep wind speed forecasting based on a data processing strategy and an optimized artificial intelligence algorithm," *Appl. Energy*, vol. 160, pp. 87–100, Oct. 2018.

[77] Q. Han, F. Meng, T. Hu, and F. Chu, "Non-parametric hybrid models for wind speed forecasting," *Energy Convers. Manag.*, 2017.

[78] H. Liu, Z. Duan, F. ze Han, and Y. fei Li, "Big multi-step wind speed forecasting model based on secondary decomposition, ensemble method and error correction algorithm," *Energy Convers. Manag.*,

2018.

[79] M. A. Chitsazan, M. Sami Fadali, and A. M. Trzynadlowski, "Wind speed and wind direction forecasting using echo state network with nonlinear functions," *Renew. Energy*, 2019.

[80] D. Y. Hong, T. Y. Ji, M. S. Li, and Q. H. Wu, "Ultra-short-term forecast of wind speed and wind power based on morphological high frequency filter and double similarity search algorithm," *Int. J. Electr. Power Energy Syst.*, 2019.

[81] H. Liu, X. Mi, and Y. Li, "Smart multi-step deep learning model for wind speed forecasting based on variational mode decomposition, singular spectrum analysis, LSTM network and ELM," *Energy Convers. Manag.*, 2018.

[82] Y. Li, H. Shi, F. Han, Z. Duan, and H. Liu, "Smart wind speed forecasting approach using various boosting algorithms, big multi-step forecasting strategy," *Renew. Energy*, 2019.

[83] J. Liu and E. Zio, "SVM hyperparameters tuning for recursive multi-step-ahead prediction," *Neural Comput. Appl.*, 2017.

[84] J. Liu and E. Zio, "A SVR-based ensemble approach for drifting data streams with recurring patterns," *Appl. Soft Comput. J.*, 2016.

[85] Z. Qu, K. Zhang, W. Mao, J. Wang, C. Liu, and W. Zhang, "Research and application of ensemble forecasting based on a novel multi-objective optimization algorithm for wind-speed forecasting," *Energy Convers. Manag.*, 2017.

[86] J. Hu, J. Wang, and L. Xiao, "A hybrid approach based on the Gaussian process with t-observation model for short-term wind speed forecasts," *Renew. Energy*, 2017.

[87] L. Xiao, F. Qian, and W. Shao, "Multi-step wind speed forecasting based on a hybrid forecasting architecture and an improved bat algorithm," *Energy Convers. Manag.*, 2017.

[88] P. Du, J. Wang, Z. Guo, and W. Yang, "Research and application of a novel hybrid forecasting system based on multi-objective optimization for wind speed forecasting," *Energy Convers. Manag.*, 2017.

[89] J. Wang, P. Du, T. Niu, and W. Yang, "A novel hybrid system based on a new proposed algorithm—Multi-Objective Whale Optimization Algorithm for wind speed forecasting," *Appl. Energy*, 2017.

[90] X. wei Mi, H. Liu, and Y. fei Li, "Wind speed forecasting method using wavelet, extreme learning machine and outlier correction algorithm," *Energy Convers. Manag.*, 2017.

[91] D. Wang, H. Luo, O. Grunder, and Y. Lin, "Multi-step ahead wind speed forecasting using an improved wavelet neural network combining variational mode decomposition and phase space reconstruction," *Renew. Energy*, 2017.

[92] M. Feurer and F. Hutter, "Hyperparameter Optimization," 2019.

[93] R. Kohavi and G. H. John, "Automatic Parameter Selection by Minimizing Estimated Error," in *Machine Learning Proceedings 1995*, 1995.

[94] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Advances in Neural Information Processing Systems*, 2012.

[95] G. Melis, C. Dyer, and P. Blunsom, “On the state of the art of evaluation in neural language models,” in *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018.

[96] J. Bergstra, D. Yamins, and D. D. Cox, “Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures,” in *30th International Conference on Machine Learning, ICML 2013*, 2013.

[97] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kegl, “Algorithms for Hyper-Parameter Optimization,” in *Advances in Neural Information Processing Systems (NIPS)*, 2011, vol. 24, p. 2546.

[98] D. Ramachandram, M. Lisicki, T. J. Shields, M. R. Amer, and G. W. Taylor, “Bayesian optimization on graph-structured search spaces: Optimizing deep multimodal fusion architectures,” *Neurocomputing*, 2018.

[99] J. Bergstra and D. Cox, “Hyperparameter Optimization and Boosting for Classifying Facial Expressions: How good can a ‘Null’ Model be?,” in *International Conference on Machine Learning*, 2013.

[100] L. F. Rodrigues, M. C. Naldi, and J. F. Mari, “Comparing convolutional neural networks and preprocessing techniques for HEp-2 cell classification in immunofluorescence images,” *Comput. Biol. Med.*, Nov. 2019.

[101] S. F. Chevtchenko, R. F. Vale, V. Macario, and F. R. Cordeiro, “A convolutional neural network with feature fusion for real-time hand posture recognition,” *Appl. Soft Comput. J.*, 2018.

[102] J. Lago, F. De Ridder, P. Vrancx, and B. De Schutter, “Forecasting day-ahead electricity prices in Europe: The importance of considering market integration,” *Appl. Energy*, 2018.

[103] J. Lago, K. De Brabandere, F. De Ridder, and B. De Schutter, “Short-term forecasting of solar irradiance without local telemetry: A generalized model using satellite data,” *Sol. Energy*, 2018.

[104] R. Mohammadi, Q. He, F. Ghofrani, A. Pathak, and A. Aref, “Exploring the impact of foot-by-foot track geometry on the occurrence of rail defects,” *Transp. Res. Part C Emerg. Technol.*, 2019.

[105] K. Kang and H. Ryu, “Predicting types of occupational accidents at construction sites in Korea using random forest model,” *Saf. Sci.*, 2019.

[106] Y. Gal and Z. Ghahramani, “Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning,” *arXiv e-prints*, p. arXiv:1506.02142, Jun. 2015.

[107] G. Corredera, M. Alves-Vieira, and O. de Bouvier, “Fouling and TSP Blockage of Steam Generators on EDF Fleet : Identified Correlations with Secondary Water Chemistry and planned Remedies,” in *International Conference, Water chemistry of nuclear reactor systems; 2008; Berlin, Germany*, 2008.

[108] S. Girard, *Physical and Statistical Models for Steam Generator Clogging Diagnosis*. Cham: Springer International Publishing, 2014.

[109] P. Luca, “Development of Unsupervised and Semi-Supervised Clustering-Based Methods for Degradation Assessment of Nuclear Power Plant Steam Generators,” Politecnico Di Milano, 2018.

[110] M. B. Kennel, R. Brown, and H. D. I. Abarbanel, “Determining embedding dimension for phase-space

reconstruction using a geometrical construction," *Phys. Rev. A*, 1992.