IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 28, 2019, accepted December 8, 2019, date of publication December 17, 2019,
date of current version December 27, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2960406

Enhanced Online Sequential Parallel Extreme Learning Machine and Its Application in Remaining Useful Life Prediction of Integrated Modular Avionics

ZEHAI GAO[1], CUNBAO MA[2], JIANFENG ZHANG[1], AND WEIJUN XU[3]
[1]School of Water Resources and Hydropower, Xi'an University of Technology, Xi'an 710048, China
[2]School of Aeronautics, Northwestern Polytechnical University, Xi'an 710072, China
[3]Development Department of Device OS, Huawei Consumer Business Group, Xi'an 710075, China

Corresponding author: Zehai Gao (gaozehai@xaut.edu.cn)

ABSTRACT Integrated modular avionics is one of the most advanced systems which has been widely applied in modern aircraft. The performance of integrated modular avionics deeply impacts flight mission. Remaining useful life prediction is the critical manner which can efficiently improve the safety and reliability of aircraft. Since integrated modular avionics is a real-time system, the prediction algorithm should have fast learning speed to satisfy the real-time requirement. In this paper, an enhanced online sequential method is proposed to predict the remaining useful life of integrated modular avionics. Firstly, this paper proposes an online sequential network which is based on the architecture of parallel layer network. Secondly, to enhance the learning capability, this paper adopts the extreme learning machine denosing autoencoder to determine the input weights of the network. Thirdly, an adaptive forgetting factor is added to further improve the performance of the proposed method. The effectiveness and the superiority of the proposed method are verified in comparison with three online sequential algorithms on the standard datasets. Finally, a degradation model is built to depict the deteriorated process of integrated modular avionics. The prediction results confirms that the proposed method can effectively address remaining useful life of integrated modular avionics.

INDEX TERMS Integrated modular avionics, parallel layer network, remaining useful life, degradation model.

I. INTRODUCTION

Integrated modular avionics (IMA) is a high-integrity, partitioned and shared computing platform, which can effectively improve the efficiency of system and reduce source consumption [1]. As a generalized and flexible hardware platform, IMA hosts almost all the avionics functions with different criticalities [2]. An Arinc653 based IMA is controlled by a real-time computer network in which time partitioning and space partitioning are implemented strictly. All the functions are executed according to partition scheduling, successively. Hence, the safety and reliability of the aircraft greatly rely on the working performance of IMA.

The associate editor coordinating the review of this manuscript and approving it for publication was Jihwan P. Choi.

Resulting from the increasing scale of integration and complex interaction between functions and resources, a new series of problems arise, which brings new requirements for the security of avionics systems [3]. Many researches are committed to guarantee the safety of IMA by virtue of structural optimization and redundancy design [4], [5]. However, quite few studies focus on the remaining useful life (RUL) prediction of IMA. The failure of IMA will affect flight mission, even cause catastrophic accidents. Accurate and timely RUL prediction can ensure flight safety and reduce the huge cost due to routine maintenance.

RUL prediction is an important part of prognostics and health management. Wang *et al.* [6] built a degradation model of aviation axial piston pump. Expectation maximization algorithm combined with Kalman filter method was used for RUL prediction. In [7], Bayesian inference using a random

walk method was implemented to predict the RUL of an aircraft fuselage panel under repeated pressurization cycles. Different from the conventional statistical approaches, many studies prefer intelligence approaches for RUL prediction due to the outstanding regression capability. In [8], the deep conventional neural network was applied to predict the RUL of rotating bearing. Zhao *et al.* [9] developed a novel method, which combined feature vector selection with support vector regression to predict the RUL of lithium-ion battery. Kamran *et al.* [10] developed a prognostics method by integrating extreme learning machine and fuzzy clustering, where the method was implemented to predict the RUL of turbofan engines. Many researches point out that the intelligence approaches perform an excellent capability in RUL prediction, especially the deep learning machines [11], [12]. Despite the success, deep learning machine still has some drawbacks. The salient disadvantages are that deep learning machine will spend a lot of time training the network and require large memory to store the parameters. Such disadvantages cannot be permitted on a real-time online prediction system. Under such circumstance, a fast and accurate online prediction method should be developed.

Extreme learning machine (ELM) was firstly proposed by Huang *et al.* [13] in 2006 as a fast and effective machine learning algorithm. Different from other single hidden layer feedforward neural networks, ELM adopts the random feature mapping to generate the hidden node parameters and uses the least square method to determine the output weights. Huang *et al.* [14] demonstrated that ELM can approximate any continuous functions and guarantee fast learning speed. Such revolution greatly reduces the time consuming, since parameters of ELM need not to be adjusted by the iterative algorithm. Many researchers focus on improving the performance of ELM. Pruned ELM and Incremental ELM are developed for optimizing the number of hidden nodes of basic ELM [15], [16]. To enhance the stability, many studies devote to determining the input weights and bias of ELM by using optimization algorithms [17], [18]. For satisfying the requirement of online testing, online sequential ELM is developed to train the data chunk-by-chunk [19]. Some studies proposed the new structures to further improve the performance of ELM [20], [21].

Inspired by the characteristics of ELM, this paper proposes an enhanced Online Sequential Parallel Extreme Learning Machine (EOS-PELM) for RUL prediction. In this paper, the architecture of parallel layer extreme learning machine is selected to construct the proposed method, in which two hidden layers with the different activation functions learn the heterogeneous features from the inputs. The weights between the input layer and the hidden layer are determined by ELM denosing autoencoder. An adaptive forgetting factor is utilized to improve the generalization ability of the proposed method. Finally, an IMA degradation model is built based on the manifested soft faults. EOS-PELM is applied to predict the RUL of IMA, and the results illustrated the effectiveness and superiority of the proposed method.

The rest of the paper is organized as follows. The basic ELM and parallel layer ELM are introduced in section 2. Section 3 describes the proposed algorithm EOS-PELM, and details the procedure of EOS-PELM for RUL prediction. Section 4 designs an IMA degradation model. In section 5, the effectiveness and superiority of EOS-PELM is validated on benchmark datasets, and the RUL prediction results demonstrate the learning performance of the proposed method. Section 6 summarizes the presented research and the future work.

II. RELATED WORK

A. EXTREME LEARNING MACHINE

ELM is an important branch of single hidden layer feedforward neural network, in which the parameters of hidden layer need not to be tuned. According to the ELM theory, input weights and bias of the network can be randomly initialized [22]–[24]. The only necessary for optimization is the output weights. A training data set with n instances is defined as $S = \left\{(x_i, y_i) \,|x_i \in R^N, y_i \in R^M, i = 1, 2, \cdots, n\right\}$, where x_i and y_i are the input vector and the output vector of the ith instance, respectively. N is the number of input attributes, and M is the dimension of the target. The relationship between the input samples and the outputs is defined as follows.

$$y_i = \beta_t \cdot f\left(w_k \cdot x_i + b_k\right) \tag{1}$$

where w_k is the input weight between the inputs and the kth hidden neuron; b_k refers to the bias of the hidden layer; β_t is the output weight between the hidden neurons and the tth output node; L represents number of hidden neurons; $f(\cdot)$ is any nonconstant piecewise continuous function. The hidden layer can be written in matrix form as follows.

$$H = \begin{bmatrix} f\left(w_1 \cdot x_1 + b_1\right) & \cdots & f\left(w_L \cdot x_1 + b_L\right) \\ \vdots & \ddots & \vdots \\ f\left(w_1 \cdot x_n + b_1\right) & \cdots & f\left(w_L \cdot x_n + b_L\right) \end{bmatrix} \tag{2}$$

The target can be defined as matrix $Y = [y_1, y_2, \cdots, y_n]^T$ and the output weights can be depicted as $\beta = [\beta_1, \beta_2, \cdots, \beta_L]^T$. In view of the ELM theory, the goal can be formulated as the following optimization problem.

$$minimize\ J_{ELM} = \frac{1}{2}\|\beta\|^2 + \frac{1}{2}\left\|Y - \hat{Y}\right\|^2. \tag{3}$$

where $\hat{Y}$ is the practical output of ELM. The output weights can be calculated by the following equation [25]–[27].

$$\beta = H^+ Y \tag{4}$$

where H^+ denotes the Moore-Penrose generalized inverse of matrix H. H^+ can be calculated as follows.

$$H^+ = \left(H^T H\right)^{-1} H^T \tag{5}$$

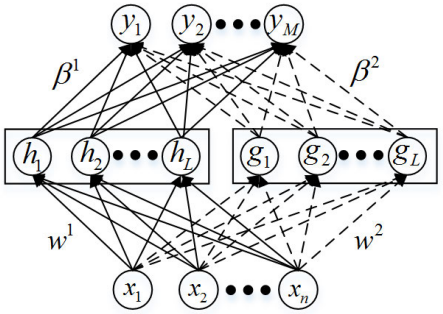

FIGURE 1. The structure of parallel layer extreme learning machine.

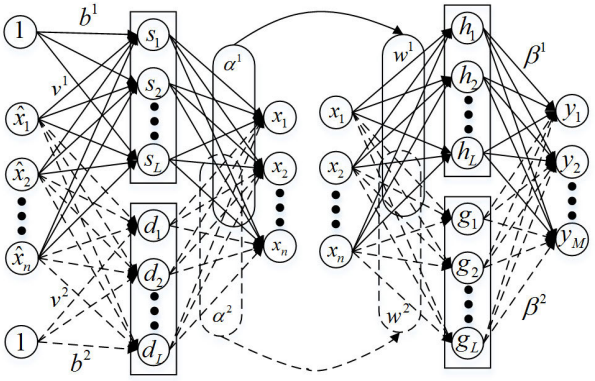

FIGURE 2. The structure of the EOS-PELM.

B. PARALLEL LAYER EXTREME LEARNING MACHINE

Parallel layer extreme learning machine is proposed based on the basic ELM. The difference with the ELM is that the input layer is mapped to two different hidden layers simultaneously. Figure 1 shows the structure of parallel layer extreme learning machine [28].

The weights between the input layer and two hidden layers are defined as w_1 and w_2. Two hidden layers are defined as $H_i = [h_{i1}, h_{i2}, \cdots, h_{iL}]$ and $G_i = [g_{i1}, g_{i2}, \cdots g_{iL}]$. Other parameters are defined as the same as the section of ELM. The relationships between the input layer and two hidden layers can be described as follows.

$$H_i = f\left(w^1 \cdot x_i + b^1\right)$$
$$G_i = f\left(w^2 \cdot x_i + b^2\right)$$

The objective function of parallel layer extreme learning machine is equivalent to the optimization problem of ELM. Then, the output can be formulated as follows.

$$y_i = \begin{bmatrix} \beta^1 & \beta^2 \end{bmatrix} \begin{bmatrix} H_i \\ G_i \end{bmatrix}$$

The output weights $\beta = \begin{bmatrix} \beta^1 & \beta^2 \end{bmatrix}$ can be determined by minimum norm least square solution. For simplifying the formulation, the output weights can be written in matrix form as follows.

$$\beta = \begin{bmatrix} H \\ G \end{bmatrix}^+ Y = \left(\begin{bmatrix} H \\ G \end{bmatrix}^T \begin{bmatrix} H \\ G \end{bmatrix} \right)^{-1} \begin{bmatrix} H \\ G \end{bmatrix}^T Y$$

where $H = [H_1, H_2, \cdots, H_n]^T$, $G = [G_1, G_2, \cdots, G_n]^T$.

III. PROPOSED METHOD

A. THE STRUCTURE OF THE PROPOSED METHOD

To ensure the convergence, random projection should be adopts to maps the inputs into feature space according to ELM theory [29], [30]. However, the performance of the algorithm is unstable resulting from the random initialization [31], [32]. In order to figure out this issue, many papers applied the optimization method to decide the input weights of ELM. In this paper, ELM denosing autoencoder is designed to determine the stable input weights of the parallel layer network.

ELM denosing autoencoder is an evolved autoencoder algorithm for feature extraction [33]. ELM autoencoder algorithm was developed according to ELM theory and the conception of autoencoder for constructing a deep learning machine, which can extract the deep features to improve the learning capability [34]. Since autoencoder can retain the completed information of the inputs, the trained output weights of the autoencoder are more stable than the random initializaiton for feature mapping. Different from conventional manners which generally select the meta-heuristic methods for optimizing the input weights of the ELM, the ELM denosing autoencoder can obtain robust features and stable weights, while keeping fast learning speed. The determined weights are used to learn the parallel hidden layers. The structure of the proposed EOS-PELM algorithm is shown in Figure 2.

The procedure of the proposed method can be broken down into three steps. In the following article, the superscript and the subscript denote the number of variables and the number of neurons in each individual layer. In order to enable the two hidden layers to express heterogeneous features, the proposed method learns the representations via two kinds of nonlinear activation functions.

In the first step, the orthogonal random input weights v^1, v^2 and biases b^1, b^2 of the ELM denosing autoencoder project the corrupted input vector $\hat{x}$ into feature space. The original input vector x is reconstructed by the hidden layer neurons s_i and d_i. The output weights α^1 and α^2 can be calculated in the same manner as ELM.

In the second step, the transposed matrix $\left(\alpha^1\right)^T$ and $\left(\alpha^2\right)^T$ are used as the input weights w^1 and w^2 to map the original input vector x into feature spaces. The hidden layers can retain completed information and stable feature of the input data by using the determined input weights.

In the third step, two hidden layers $H_i = [h_{i1}, h_{i2}, \cdots, h_{iL}]$ and $G_i = [g_{i1}, g_{i2}, \cdots g_{iL}]$, which are transformed by two different activation functions, are used to determine the final output weights $\beta = \begin{bmatrix} \beta^1 & \beta^2 \end{bmatrix}$. The output can be got in

matrix form as $\hat{Y} = \left[\beta^1 \ \beta^2\right]\left[H \ G\right]^T$. The proposed method still has a fast learning speed for which the parameters need not to be adjusted iteratively.

B. ONLINE LEARNING WITH FORGETTING FACTOR

As mentioned in the above section, the parameters of EOS-PELM are initialized by the first batch of input samples. In the online stage, supposed that the kth batch of input samples is $T_k = \{(x_i, y_i)\}_{i=1+\sum_{j=0}^{k-1} N_j}^{\sum_{j=0}^{k} N_j}$, where N_j denotes the number of instances in the jth batch. The output weight is $\beta_k = \left[\beta_k^1 \ \beta_k^2\right]$ and the hidden layers are H_k and G_k. Then the new arrived batch of input samples will generate the new hidden layers H_{k+1} and G_{k+1}. The output weight can be acquired by solving the following function.

$$\beta_{k+1} = \arg\min \left\| \beta_{k+1} \begin{bmatrix} \lambda H_k & H_{k+1} \\ \lambda G_k & G_{k+1} \end{bmatrix} - \begin{bmatrix} \lambda Y_k & Y_{k+1} \end{bmatrix} \right\|$$

where λ is the forgetting factor. According to ELM theory, the target can be approximated by using random projection and least square method. Then, β_{k+1} can be calculated as follows.

$$\begin{aligned} \beta_{k+1} &= \begin{bmatrix} \lambda Y_k & Y_{k+1} \end{bmatrix} \begin{bmatrix} \lambda H_k & H_{k+1} \\ \lambda G_k & G_{k+1} \end{bmatrix}^T \\ &\quad \cdot \left[\begin{bmatrix} \lambda H_k & H_{k+1} \\ \lambda G_k & G_{k+1} \end{bmatrix} \begin{bmatrix} \lambda H_k & H_{k+1} \\ \lambda G_k & G_{k+1} \end{bmatrix}^T \right]^{-1} \\ &= \begin{bmatrix} \lambda Y_k & Y_{k+1} \end{bmatrix} \begin{bmatrix} \lambda Z_k \\ Z_{k+1} \end{bmatrix} \left[\begin{bmatrix} \lambda Z_k & Z_{k+1} \end{bmatrix} \begin{bmatrix} \lambda Z_k \\ Z_{k+1} \end{bmatrix} \right]^{-1} \\ &= \left[\lambda^2 Y_k Z_k^T + Y_{k+1} Z_{k+1}^T \right] \left[\lambda^2 Z_k Z_k^T + Z_{k+1} Z_{k+1}^T \right]^{-1} \end{aligned}$$

where $Y_{k+1} = \left[y_{N_k+1}, y_{N_k+2}, \cdots y_{N_{k+1}}\right]^T$ is the output vector, and the subscript of the capitals denotes the number of the batch.

The defined matrix K_{k+1} can be formulated as follows.

$$\begin{aligned} K_{k+1} &= \begin{bmatrix} \lambda Z_k & Z_{k+1} \end{bmatrix} \begin{bmatrix} \lambda Z_k^T \\ Z_{k+1}^T \end{bmatrix} \\ &= \lambda^2 Z_k Z_k^T + Z_{k+1} Z_{k+1}^T \\ &= \lambda^2 K_k + Z_{k+1} Z_{k+1}^T \end{aligned}$$

It can be deduced that

$$\begin{aligned} &\lambda^2 Y_k Z_k^T + Y_{k+1} Z_{k+1}^T \\ &\qquad = \lambda^2 \beta_k K_{k+1} + \left(Y_{K+1} - \lambda^2 \beta_k Z_{k+1} \right) Z_{k+1}^T \end{aligned}$$

Then, β_{k+1} can be transformed as follows.

$$\begin{aligned} \beta_{k+1} &= \left[\lambda^2 \beta_k K_{k+1} + \left(Y_{k+1} - \lambda^2 \beta_k Z_{k+1} \right) Z_{k+1}^T \right] K_{k+1}^{-1} \\ &= \lambda^2 \beta_k + \left(Y_{k+1} - \lambda^2 \beta_k Z_{k+1} \right) Z_{k+1}^T K_{k+1}^{-1} \end{aligned}$$

There exists the matrices $P_k = K_k^{-1}$, $P_{k+1} = K_{k+1}^{-1}$. So,

$$P_{k+1} = \frac{1}{\lambda^2} \left(P_k - \frac{P_k Z_{k+1} Z_{k+1}^T P_k}{\lambda^2 I + Z_{k+1}^T P_k Z_{k+1}} \right)$$

TABLE 1. General activation functions.

Function names	Formulations
Sigmoid	$f(x) = 1/\left(1 + e^{-x}\right)$
Gaussian (Radial basis function)	$f(x) = e^{-x^2}$
Rectified linear unit (ReLu)	$f(x) = \begin{cases} x, x \geq 0 \\ 0, x < 0 \end{cases}$
TanH	$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
Hardlim	$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$
Sine	$f(x) = \sin(x)$
Triangular basis function	$f(x) = \begin{cases} 1 - \lvert x \rvert, -1 \leq x \leq 1 \\ 0, otherwise \end{cases}$

Finally, the updating function of output weight can be formulated as follows.

$$\beta_{k+1} = \beta_k + \left(Y_{k+1} - \beta_k Z_{k+1}\right) Z_{k+1}^T P_{k+1}$$

If the forgetting factor is set as $\lambda = 1$, the former data will be remembered. If $\lambda = 0$, the former data will be aborted. The forgetting factor therefore plays a critical role in online learning [35]. In this paper, an adaptive method is presented to adjust the forgetting factor. Since $0 \leq \lambda \leq 1$, the root mean square error (RMSE) E_k which is with respect to the learning accuracy of old model is used to build the forgetting factor. The equation of the current forgetting factor is defined as follows.

$$\lambda_{k+1} = \exp\left(-E_k\right)$$

ELM using a nonlinear piecewise continuous function as the activation function $f(\cdot)$ can approximate any continuous target function Y. The general activation functions are listed in Table 1. Sigmoid function is employed to almost all kinds of neural networks as the typically activation function. ReLu and its extension functions are developed recently in deep learning field, which perform excellent capability in many pattern recognition problems [36], [37].

The neural networks with different activation functions usually show different characteristics and complementary learning behaviors. In order to overcome the limitations of the neural network with individual activation function and enhance the generalization performance, the proposed method adopts different kinds of activation functions for the hidden layers. The learning performance of the proposed method can be improved by virtue of mapping the raw data into heterogeneous feature spaces.

C. GENERAL PROCEDURE OF THE PROPOSED METHOD

In this paper, an enhanced online sequential method with fast learning speed is developed for RUL prediction. The flowchart of proposed method is shown in Figure 3 and the general procedure for RUL prediction is summarized as follows.

Step 1. The running data sets are collected chunk-by-chunk.

Step 2. The first batch of data is applied to initialize the proposed method.

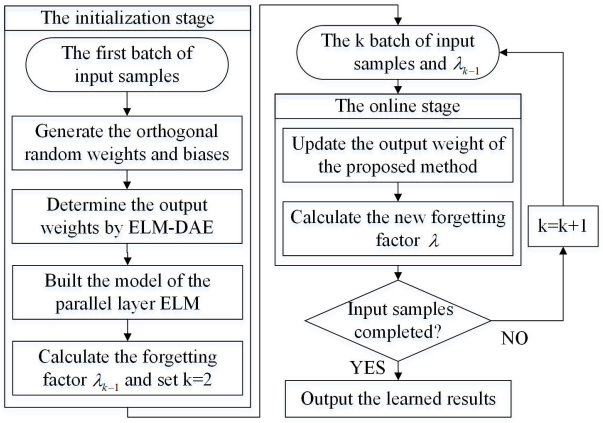

FIGURE 3. The flowchart of the proposed method.

Step 3. The following batches are used for adjusting the parameters of the model based on the online learning algorithm.

Step 4. The new arrived data sets are defined as the inputs for RUL prediction.

Step 5. Report the prediction results.

IV. IMA DEGRADATION MODEL

IMA is an electronic equipment with many modules, such as, general processor module, figure processor module, power module and common data network [38]. Therefore, IMA exists lots of components which will deteriorate during the service. The degradation factors won't lead to a failure directly, but the factors will cause a series of errors and affect the working efficiency. This paper concentrates on the working efficiency for further RUL prediction.

According to the ARINC653 standard, a health monitoring function is designed to address the errors for guaranteeing the safety of IMA. Since error handling will occupy the time of the faulty function, IMA assigns the reasonable running time slice to the hosted functions. Soft fault denotes the fault that can be recovered during the running time. On this basis, this paper describes the performance of IMA with the characteristics of the manifested soft faults.

Many reasons may result in soft faults, such as, extreme environment, extreme vibration and electronic devices degradation. In terms of the civil aviation, extreme working condition is rarely encountered. However, the electronic devices degradation cannot be neglected. Many researches have pointed out that almost all the factors for soft faults, for instance, time-dependent dielectric breakdown (TDDB), hot carrier injection (HCI) and electromigration (EM), are caused by wearout resulting from extensive use [39]. Soft faults happen in a low frequency during the initial stage, which are identified as small noise fluctuation. As time goes by, the frequency and duration increase, and soft faults start to occur. In the final stage, soft faults will seriously impact on the system, even can lead to a failure [40]. Figure 4 shows the frequency of soft faults in the whole life cycle of the electronic system.

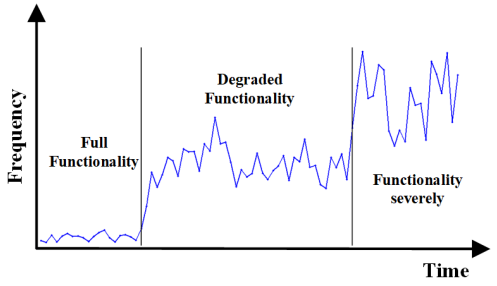

FIGURE 4. The frequency of soft faults in the whole life cycle.

As a core computing and real-time system, the computing accuracy and the computing speed are the basic requirements of IMA. However, computing accuracy heavily depends on the data collection which is separated from the IMA. Hence, real-time requirement is the main criterion to assess the performance of IMA.

In this paper, a novel IMA degradation model is built under Lévy Process [41]. Every basic function in IMA will be assigned a fixed time slice for completing the operation. The assigned time slice is a litter longer than the practical function execution time, the redundancy is designed to process some issues, such as error handling, during the running time. In other words, the function completion time $X(t)$ consists of function execution time and the time delay caused by error handling. The function should be completed in the assigned time slice, otherwise the function will fail. Suppose that the performance of IMA is constant in a fly cycle, thenthe following properties of the function completion time $X(t)$ can be proved in the allocated time slice. Firstly, the initial time $X(0) = 0$. Secondly, it has stationary increments and independent increments. Hence, the function completion time $X(t)$ can be depicted using Lévy Process as follows [42].

$$X(t) = \mu t + B(t) + \sum_{i=0}^{N(t)} W_i, \quad N(t) = 0, 1, 2 \ldots.$$

where $\mu t + B(t)$ is a Wiener process, whose two parts denote the function execution time without soft faults and the random error respectively. $\sum_{i=0}^{N(t)} W_i$ is a compound Poisson process. W_i is the time delay caused by the ith error handling; $N(t)$ denotes the total number of soft faults by time t. Based on the divisibility of Poisson process, the function completion time $X(t)$ can be illustrated as follows.

$$\begin{aligned} X(t) &= \mu t + B(t) + \sum_{j=1}^{n} \sum_{i=1}^{N_j(t)} W_i \\ &= \mu t + B(t) + \sum_{i=1}^{N_1(t)} W_i + \sum_{i=1}^{N_2(t)} W_i + \cdots + \sum_{i=1}^{N_n(t)} W_i \\ &= \mu t + B(t) + \sum_{i=1}^{E(N_1(t)|\lambda_1(T))} W_i + \sum_{i=1}^{E(N_2(t)|\lambda_2(T))} W_i \\ &\quad + \cdots + \sum_{i=1}^{E(N_n(t)|\lambda_n(T))} W_i \end{aligned}$$

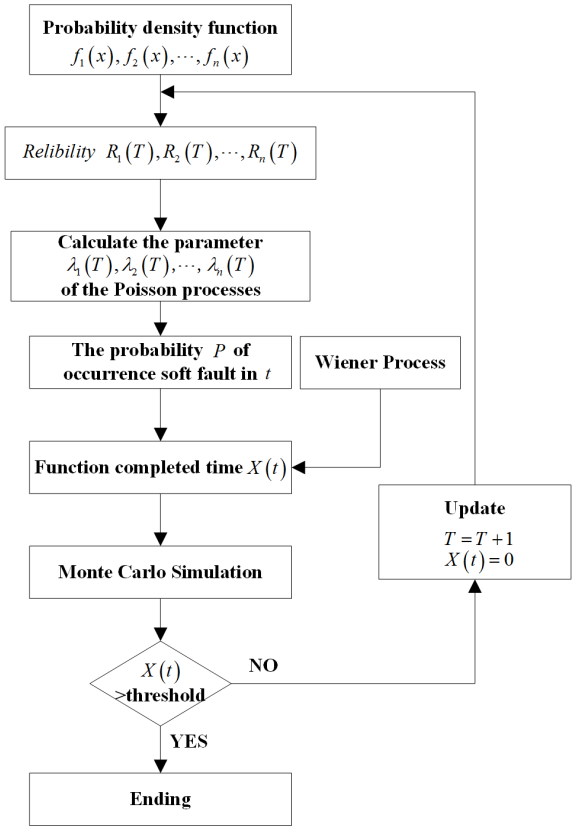

FIGURE 5. The simulation flowchart.

where $E\left(N_j(t)\middle|\lambda_j(T)\right)$ denotes the expectation of the soft faults caused by the jth factor under the parameter $\lambda_j(T)$ in the Tth fly cycle.

As mentioned above, TDDB, HCI and EM are the typical degradation reasons responsible for electronic devices, which should be considered in IMA. Many researches devote to revealing the mechanism of TDDB, HCI and EM. The significant conclusions are summarized based on lots of experimental results. The degradation of electronics caused by TDDB and HCI can be described by Weibull distribution [43]. The degradation of electronics caused by EM can be described by Lognormal distribution [44]. The reliability of the electronic devices in IMA which is effected by single factor can be calculated as follows.

$$R(T) = P(X > T) = \int_T^{\infty} f(x)dx$$

where $f(x)$ is probability density function. Hence, multiple factors correspond to multiple probability density functions $f_1(x)f_2(x)\cdots f_n(x)$ which correspond to multiple reliabilities $R_1(T), R_2(T), \cdots, R_n(T)$. Meanwhile, $P(X > t) = P(k = 1) = \frac{(\lambda t)^k}{k!}e^{-\lambda t} = \lambda t e^{-\lambda t}$ implies no soft fault until time t. Hence, there exists a relationship between $R_j(T)$ and $\lambda_j(T)$, and it can be defined as follows.

$$\lambda_j(T) = C \cdot \sqrt{\frac{1}{R_j(T)}}$$

FIGURE 6. The Freescale P2020RDB.

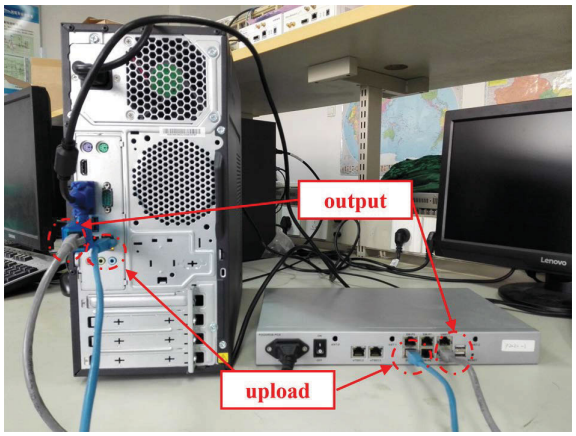

FIGURE 7. The connection fashion.

where C is coefficient. The occurrence of soft fault in a fly cycle can be depicted using homogeneous Poisson process. So, $E\left(N_j(t)\middle|\lambda_j(T)\right)$ can be transformed as follows.

$$\begin{aligned} E\left(N_j(t)\middle|\lambda_j(T)\right) &= E\left(N_j(t)\middle|R_j(T)\right) \\ &= E\left(N_j(t)\middle|f_j(x), T\right) \end{aligned}$$

The parameters of the Wiener process and the error handing time W_i can be obtained by the experiments on VxWorks. After setting total number of fly cycles and defining threshold, namely, the time slice, the Monte Carlo method is applied to simulate the degradation process of IMA. Figure 11 shows the flowchart of simulation.

This paper builds an experimental platform. Figure 6 is a Freescale P2020RDB (Reference Design Board). The configuration and programs are built on the computer and uploaded to the P2020RDB by Gigabit Ethernet. The running results are monitored by computer via another Gigabit Ethernet terminal. Figure 7 shows the connection fashion. The configuration and functions of IMA are designed by VxWorks.

As a real-time system, all the functions of IMA should be finished in the stipulated time slices [45]. If an error can be amended in time, health monitor function only sends a message to the log. The maintainer will define the error as

TABLE 2. Basic information of benchmark datasets.

Datasets	No. of Attributes	No. of Instances	Training Instances	Testing Instances
servo	4	167	120	47
autoMPG	8	392	320	72
concrete	8	1030	900	130
abalone	77	4177	3000	1177
winequality	11	4898	4000	898
Superconductivty	81	21263	15000	6263

TABLE 3. Structures, batch sizes and dropout rates.

Datasets	Structures	Batch sizes	Dropout rates
servo	$4-\{12,12\}-1$	30	0.1
autoMPG	$8-\{15,15\}-1$	40	0.1
concrete	$8-\{20,20\}-1$	150	0.1
abalone	$7-\{25,25\}-1$	300	0.2
winequality	$11-\{10,10\}-1$	1000	0.2
Superconductivty	$81-\{25,25\}-1$	1000	0.4

a soft fault. If the function cannot complete in the stipulated time, IMA will regard this condition as failure.

V. EXPERIMENTS AND RESULTS

A. BENCHMARKS

Benchmark datasets employed in this study are servo, autoMPG, concrete, abalone, winequality and Superconductivty datasets. All the benchmark datasets are collected from UCI Machine Learning Repository [46]. The number of attributes involved in the regression problem, number of training instances and testing instances of each dataset are listed in Table 2.

In this paper, Sigmoid function and ReLu function are employed as activation functions for EOS-PELM. In order to validate the learning performance, the proposed algorithm is compared with three online prediction methods. The compared methods are online parallel layer extreme learning machine (PL-ELM), online sequential extreme learning machine (OS-ELM) and online sequential fast learning machine [47], [48]. For ensuring the learning efficiency of these four algorithms, the number of hidden neurons of these algorithms satisfies the following constraints. The number of hidden neurons of PL-ELM is equivalent to EOS-PLEM. The number of hidden neurons of OS-ELM is twice as much as the defined number of hidden neurons of the proposed method. The number of hidden neurons of fast learning machine is equal to the number of hidden neurons of EOS-PLEM. The structures, batch sizes and dropout rates of the proposed method for the benchmark datasets are listed in Table 3.

In this paper, each algorithm is repeated 50 times to obtain credible results. The training MAE, RMSE, the training standard deviations (Std) and the training time (in seconds) are listed in Table 4. The testing MAE, RMSE, the testing Std and the training time (in milliseconds) are presented in Table 5.

Figure 8-13 show the testing error $E = Y - \hat{Y}$ on different datasets, where $\hat{Y}$ is the practical output of the networks and Y is the target. To make it clear, Figure 11, 12 and 13 show the errors of testing results after subsampling while the sampling interval are set as 10, 10, and 20, respectively.

TABLE 4. The training results.

		EOS-PELM	PL-ELM	OS-ELM	OS-FLN
servo	MAE	0.0693	0.0764	0.0905	0.0892
	RMSE	0.1501	0.1364	0.1672	0.1783
	Std	0.0116	0.0144	0.0132	0.0141
	Time(s)	0.0039	0.0036	0.0008	0.0034
autoMPG	MAE	0.0369	0.0373	0.0456	0.0399
	RMSE	0.0943	0.0878	0.1091	0.1097
	Std	0.0023	0.0020	0.0031	0.0036
	Time(s)	0.0042	0.041	0.0021	0.0036
concrete	MAE	0.0712	0.0716	0.0906	0.0792
	RMSE	0.1693	0.2037	0.2005	0.3472
	Std	0.0072	0.0175	0.0077	0.5254
	Time(s)	0.0041	0.0039	0.0020	0.0038
abalone	MAE	0.0536	0.0534	0.0549	0.0557
	RMSE	0.139	0.1376	0.1503	0.1589
	Std	0.0011	0.0017	0.0021	0.0306
	Time(s)	0.0056	0.0044	0.0031	0.0043
winequality	MAE	0.0956	0.0974	0.0982	0.0963
	RMSE	0.1236	0.1238	0.1258	0.1244
	Std	0.0008	0.0007	0.0006	0.0005
	Time(s)	0.0068	0.0050	0.0035	0.0050
Superconductivty	MAE	0.0744	0.0752	0.0884	0.0676
	RMSE	0.0964	0.0983	0.1236	0.0895
	Std	0.0018	0.0012	0.0045	0.0008
	Time(s)	0.0096	0.074	0.0067	0.0094

TABLE 5. The testing results.

		EOS-PELM	PL-ELM	OS-ELM	OS-FLN
servo	MAE	0.0776	0.0782	0.0919	0.0922
	RMSE	0.1926	0.1983	0.1995	0.1973
	Std	0.0155	0.0241	0.0212	0.0264
	Time(ms)	0.1554	0.1553	0.0071	0.0617
autoMPG	MAE	0.0989	0.0946	0.0942	0.0918
	RMSE	0.2298	0.2653	0.2574	0.2524
	Std	0.0094	0.0378	0.0173	0.0238
	Time(ms)	0.1818	0.1822	0.1040	0.0816
concrete	MAE	0.0734	0.0740	0.0802	0.0743
	RMSE	0.1906	0.2307	0.1974	0.3064
	Std	0.0182	0.0625	0.028	0.3937
	Time(ms)	0.2280	0.2283	0.1475	0.1306
abalone	MAE	0.0538	0.0535	0.0542	0.0551
	RMSE	0.1451	0.1447	0.1454	0.152
	Std	0.0004	0.0023	0.0005	0.0295
	Time(ms)	0.9745	0.9824	0.9943	0.7004
winequality	MAE	0.0946	0.0957	0.0962	0.0959
	RMSE	0.1154	0.1162	0.1168	0.1172
	Std	0.0015	0.0017	0.0016	0.0014
	Time(ms)	0.5424	0.6131	0.7859	0.4887
Superconductivty	MAE	0.0602	0.0618	0.0715	0.0626
	RMSE	0.0815	0.0837	0.0982	0.0898
	Std	0.0042	0.0047	0.0078	0.0056
	Time(ms)	12.4	14.1	11.6	11.8

As the above tables and figures show, the proposed method has the small standard deviations in both training step and testing step which result from the stable input weights via the ELM denosing autoencoder. Although PL-ELM has better training performance in some cases, the proposed method still has a stable and good training capability for different datasets. The testing results further confirm that the proposed method outperforms other algorithms. Compared with PL-ELM, using different activation functions in hidden layers provides a better generalization performance for the proposed method. The proposed method show the outstanding learning

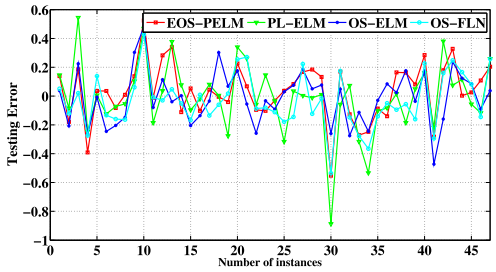

FIGURE 8. The testing errors on servo dataset.

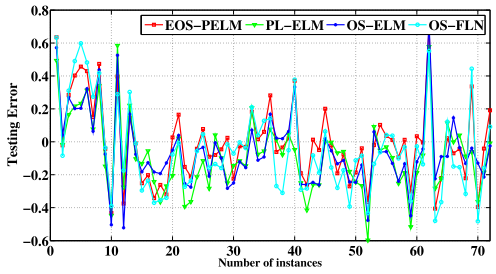

FIGURE 9. The testing errors on autoMPG dataset.

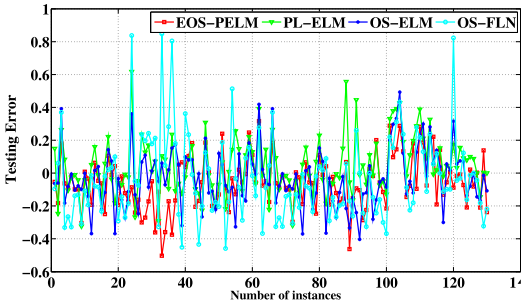

FIGURE 10. The testing errors on concrete dataset.

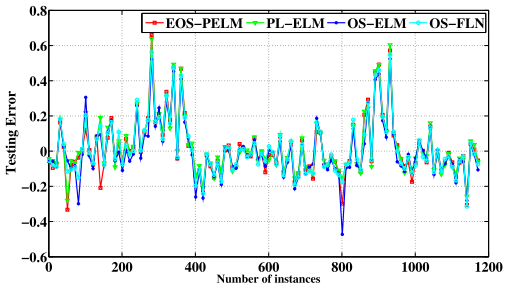

FIGURE 11. The testing errors on abalone dataset.

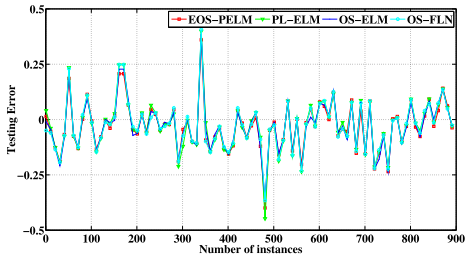

FIGURE 12. The testing errors on winequality dataset.

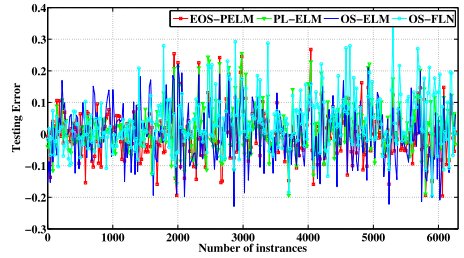

FIGURE 13. The testing errors on Superconductivty dataset.

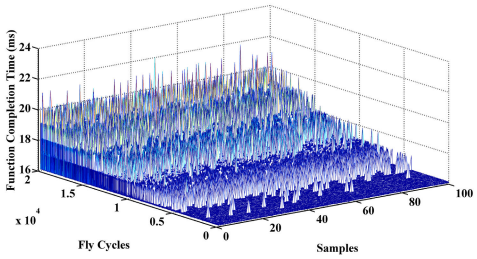

FIGURE 14. The simulation data of IMA degradation process.

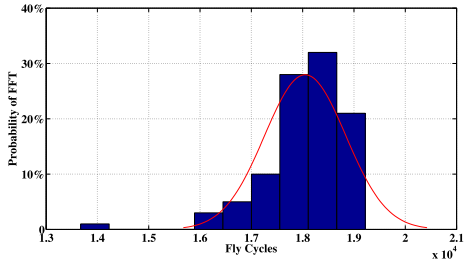

FIGURE 15. The staistics of FFT.

performance in both the small and large instances datasets. Among the four methods, although the proposed method requires more time to train, it still has the fast training speed, which enable the proposed method to suit to the real-time online prediction environment.

B. RUL PREDICTION OF IMA

This paper focuses on the degradation process from no soft fault to failure. IMA degradation performance is defined as the constant in single fly cycle, and the simulation in every fly cycle is repeated 100 times. The simulation data including 20000 fly cycles is shown in Figure 14.

In this paper, the time slice is set as 20 milliseconds. The 100 samples in every fly cycle are used as the attributes of the input data. The mean value of the function completion time is defined as the target. The statistics of the first failure time (FFT) is presented in Figure 15.

The expectation of the FFT corresponds to the 18073th fly cycle in Figure 15. The simulation data from 1 to 16600 are used as the training data, and the target data from 1001 to 176000 are set as the training label. The simulation data from

TABLE 6. The results.

		EOS-PELM	PL-ELM	OS-ELM	OS-FLN
Training results	MAE	0.0214	0.0232	0.0303	0.0201
	RMSE	0.0908	0.0573	0.1026	0.0981
	Std	0.0002	0.0015	0.0019	0.185
	Time(s)	0.6822	0.6123	0.6116	0.6297
Testing results	MAE	0.0198	0.0222	0.0248	0.0216
	RMSE	0.2274	0.2358	0.4355	0.2323
	Std	0.0272	0.0354	0.0296	0.0459
	Time(ms)	2.9	3.3	2.9	3.6
Prediction results	FFT	18087	17837	17631	17877
	RUL	487	237	31	277
	Error	14	-236	-442	-196

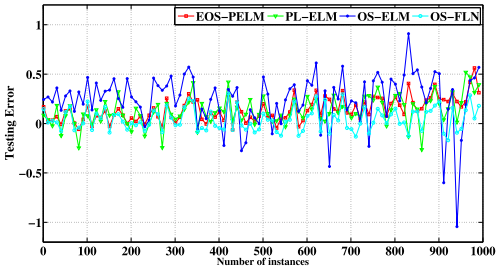

FIGURE 16. The errors of prediction results.

16601 to 18600 are used as the testing data, and the target data from 17001 to 18600 are set as the testing label. The structure of the proposed method is 100-20, 20-1, the dropout rate is 0.1, and the batch size is 200. The prediction algorithms are run on the computer which acquires the data from the P2020RDB. All the four algorithms are repeated 50 times, and the training results, testing results and prediction results are listed in Table 6.

To make it clear, figure 16 shows the prediction result errors $E = Y - \hat{Y}$ of four algorithms after subsampling and the sampling interval is set as 10, where $\hat{Y}$ is the practical output of the networks and Y is the target.

As Table 6 and Figure 16 show, the proposed method has the best testing accuracy, although its training accuracy is not outstanding. Compared with other three algorithms, the proposed method has the smallest standard deviation, which illustrates that the EOS-PELM algorithm possesses a stable performance. In terms of the time consuming, training and testing time of the proposed method takes more time than other three methods, however it still keeps the fast speed. The fast learning speed enables that EOS-PELM to be implemented to the real time system for online RUL prediction. The prediction results further demonstrate that the EOS-PELM is the best method for RUL prediction of IMA among these four methods. EOS-PELM employs the parallel hidden layer structure and ELM-DAE algorithm to learn the heterogeneous features and retain the reliable information from the raw data. Meanwhile, the adaptive online learning fashion and the conception of least square method guarantee the fast learning speed and the capability of prediction. All the advantages make the proposed method show an excellent performance in RUL prediction of IMA.

VI. CONCLUSION

In this paper, an online prediction method called EOS-PELM is proposed for RUL prediction of IMA. The proposed method adopts the structure of parallel layer network. The input parameters of the network are determined by the designed ELM denosing autoencoder which can efficiently enhance the stability. The parallel hidden layers with different kinds of activation functions is seclected for capturing the heterogeneous features, thereby improving the learning performance of the network. The output weight of the proposed method is calculated in the same manner as ELM. Besides, the proposed method adopts the adaptive online learning fashion which can promote the generalization and keep the fast learning speed. For validating the performance, the proposed method is performed on the benchmark datasets, and the results confirm that the proposed method show an excellent prediction capability as compared with three online prediction methods. This paper uses the Lévy Process to model the degradation process of IMA. The degradation model is built based on the soft faults and the simulation samples are collected for RUL prediction. The prediction results confirm that the proposed method is more stable and more accurate than the other three online prediction algorithms. The training and testing time demonstrate that the EOS-PELM still has the fast learning speed, which enables the proposed method to suit to real time system. In the future research, the practical environment with real data should be focused on.

REFERENCES

[1] X. Liu, C. Cao, X. Zhao, J. Sun, and J. Zhu, "Network performance analysis of time-triggered Ethernet based on network calculus for DIMA," in *Proc. IEEE/AIAA Digit. Avionics Syst. Conf.*, Sep. 2015, pp. 13–17.

[2] T. Robati, A. Gherbi, and J. Mullins, "A modeling and verification approach to the design of distributed IMA architectures using TT Ethernet," in *Proc. Int. Conf. Ambient Syst. Netw. Technol.*, 2016, pp. 23–26.

[3] H. Wang, T. Zhao, F. Ren, and Z. Jiang, "Integrated modular avionics system safety analysis based on model checking," in *Proc. Annu. Rel. Maintainability Symp.*, Jan. 2017, pp. 1–6.

[4] Z. Gao, C. Ma, and Y. Luo, "RUL prediction for IMA based on deep regression method," in *Proc. 10th IEEE Int. Workshop Comput. Intell. Appl.*, Nov. 2017, pp. 25–31.

[5] Z. Li, S. Wang, T. Zhao, and B. Liu, "A hazard analysis via an improved timed colored petri net with time-space coupling safety constraint," *Chin. J. Aeronaut*, vol. 29, no. 4, 1027-1041, 2016.

[6] X. Wang, S. Lin, S. Wang, Z. He, and C. Zhang, "Remaining useful life prediction based on the Wiener process for an aviation axial piston pump," *Chin. J. Aeronaut*, vol. 29, no. 3, pp. 779–788, 2016.

[7] J. M. Karandikar, N. H. Kim, and T. L. Schmitz, "Prediction of remaining useful life for fatigue-damaged structures using Bayesian inference," *Eng. Fract. Mech.*, vol. 96, pp. 588–605, Dec. 2012.

[8] L. Ren, Y. Q. Sun, H. Wang, and L. Zhang, "Prediction of bearing remaining useful life with deep convolution neural network," *IEEE Access*, vol. 6, pp. 13041–13049, 2018.

[9] Q. Zhao, X. L. Qin, H. B. Zhao, and W. Q. Feng, "A novel prediction method based on the support vector regression for the remaining useful life of lithium-ion batteries," *Microelectron. Reliab.*, vol. 85, pp. 99–108, Jun. 2018.

[10] J. Kamran, G. Rafael, and Z. Noureddine, "A new multivariate approach for prognostics based on extreme learning machine and fuzzy clustering," *IEEE Trans. Cybern.*, vol. 45, no. 12, pp. 2626–2639, Dec. 2015.

[11] Z. H. Gao, C. B. Ma, D. Song, and Y. Liu, "Deep quantum inspired neural network with application to aircraft fuel system fault diagnosis," *Neurocomputing*, vol. 238, pp. 13–23, May 2017.

[12] X. Li, Q. Ding, and J. Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Reliab. Eng. Syst. Saf.*, vol. 172, pp. 1–11, Apr. 2018.
[13] G. B. Huang and Q. Y. C. K. Zhuand Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, pp. 489–501, Dec. 2006.
[14] G.-B. Huang, Z. Bai, L. L. C. Lekamalage, and C. M. Kasun, "Local receptive fields based extreme learning machine," *IEEE Comput. Intell. Mag.*, vol. 10, no. 2, pp. 18–29, May 2015.
[15] G. B. Huang, M. B. Li, L. Chen, and C. K. Siew, "Incremental extreme learning machine with fully complex hidden nodes," *Neurocomputing*, vol. 71, pp. 576–583, Jan. 2008.
[16] H. J. Rong, Y. S. Ong, A. H. Tan, and Z. X. Zhu, "A fast pruned-extreme learning machine for classification problem," *Neurocomputing*, vol. 72, pp. 359–366, Dec. 2008.
[17] S. Suresh, S. Saraswathi, and N. Sundararajan, "Performance enhancement of extreme learning machine for multi-category sparse data classification problems," *Eng. Appl. Artif. Intel.*, vol. 23, pp. 1149–1157, Oct. 2010.
[18] M. Luo, C. Li, X. Zhang, R. Li, and X. An, "Compound feature selection and parameter optimization of ELM for fault diagnosis of rolling element bearings," *IAS Trans.*, vol. 65, pp. 556–566, Nov. 2016.
[19] X. Wang and M. Han, "Improved extreme learning machine for multivariate time series online sequential prediction," *Eng. Appl. Artif. Intell.*, vol. 40, pp. 28–36, Apr. 2015.
[20] P. A. Henríquez and G. A. Ruz, "Extreme learning machine with a deterministic assignment of hidden weights in two parallel layers," *Neurocomputing*, vol. 226, pp. 109–116, Feb. 2017.
[21] L. D. Tavares, R. R. Saldanha, and D. A. G. Vieira, "Extreme learning machine with parallel layer perceptrons," *Neurocomputing*, vol. 166, pp. 164–171, Oct. 2015.
[22] G. B. Huang, L. Chen, and C. K. Siew, "Universal approximation using incremental constructive feedforward networks with random hidden nodes," *IEEE Trans. Neural Netw.*, vol. 4, no. 17, pp. 879–892, Jul. 2006.
[23] J. Wu and C. L. Liu, "Short-term wind power prediction based on empirical mode decomposition and extreme learning machine," *J. Electr. Eng. Technol.*, vol. 13, pp. 1841–1851, 2016.
[24] Z. Tian, S. Li, Y. Wang, and Y. Sha, "A prediction method based on wavelet transform and multiple models fusion for chaotic time series," *Chaos, Soliton Fract.*, no. 98, pp. 158–172, May 2017.
[25] G. B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst., Man, Cybern. B. Cybern*, vol. 42, no. 2, pp. 513–529, Apr. 2012.
[26] Z. Tian, S. Li, Y. Wang, and X. Wang, "Network traffic prediction method based on improved ABC algorithm optimized EM-ELM," *J. China Universities Posts Telecommun.*, vol. 25, pp. 33–44, 2018.
[27] J. L. Yin Li, Y. Cao, and J. Zhao, "An adaptive online sequential extreme learning machine for short-term wind speed prediction based on improved artificial bee colony algorithm," *Neural Netw. World*, vol. 28, pp. 191–212, May 2018.
[28] Y. Q. Wang, Y. Dou, X. W. Liu, and Y. W. Lei, "PR-ELM: Parallel regularized extreme learning machine based on cluster," *Neurocomputing*, vol. 173, pp. 1073–1081, Jan. 2016.
[29] Z. Tian, G. Wang, S. Li, Y. Wang, and X. Wang, "Artificial bee colony algorithm–optimized error minimized extreme learning machine and its application in short-term wind speed prediction," *Wind Eng.*, vol. 43, pp. 263–276, Jun. 2019.
[30] Y. G. Wang, F. L. Cao, and Y. B. Yuan, "A study on effectiveness of extreme learning machine," *Neurocomputing*, vol. 74, pp. 2483–2490, Sep. 2011.
[31] J. X. Tang, C. W. Deng, and G. B. Huang, "Extreme learning machine for multilayer perceptron," *IEEE T. Neur. Net. Lear.*, vol. 27, no. 4, pp. 809–821, Apr. 2016.
[32] Z. Tian, Y. Ren, and G. Wang, "Short-term wind speed prediction based on improved PSO algorithm optimized EM-ELM," *Energy Sources, A, Recovery, Utilization, Environ. Effects*, vol. 41, no. 1, pp. 26–46, 2019.
[33] Z. H. Gao, C. B. Ma, Z. Y. She, and X. Dong, "An enhanced deep extreme learning machine for integrated modular avionics health state estimation," *IEEE Access*, vol. 6, pp. 65813–65823, 2018.
[34] L. L. C. Kasun, H. Zhou, G. B. Huang, and M. V. Chi, "Representational learning with ELMs for big data," *IEEE Intell. Syst.*, vol. 28, no. 6, pp. 31–34, 2013.
[35] H. G. Zhang, S. Zhang, and Y. Yin, "Online sequential ELM algorithm with forgetting factor for real applications," *Neurocomputing*, vol. 261, pp. 144–152, Oct. 2017.
[36] Y. Dmitry, "Error bounds for approximations with deep ReLU networks," *Neural Netw.*, no. 94, pp. 103–114, Oct. 2017.
[37] X. J. Jin, C. Y. Xu, J. S. Feng, Y. C. Wei, J. J. Xiong, and S. C. Yan, "Deep learning with S-shaped Rectified linear activation units," *Comput. Sci.*, vol. 3, pp. 1–8, Feb. 2015.
[38] M. Halle and F. Thielecke, "Model-based transition of IMA architecture into configuration data," in *Proc. IEEE/AIAA Digital Avionics Syst. Conf.*, Sep. 2016, pp. 25–29.
[39] M. Denertzi, B. Zandian, R. Rojas, and M. Annavaram, "Benchmarking ISA reliability to intermittent errors," in *Proc. Int. Symp. Workload Characterization*, 2012, pp. 86–87.
[40] A. Correcher, E. Garcia, F. Morant, E. Quiles, and L. Rodriguez, "Intermittent failure dynamics characterization," *IEEE Trans. Rel.*, vol. 61, no. 3, pp. 649–658, Sep. 2012.
[41] J. Zheng, Z. Lin, and C. Tong and R. Ye, "New methods of simulating Lévy processes," *Phys. A, Stat. Mech. Appl.*, vol. 473, pp. 461–466, May 2017.
[42] D. Hainaut, "Clustered Lévy processes and their financial applications," *J. Comput. Appl. Math.*, vol. 319, pp. 117–140, Aug. 2017.
[43] B. P. Cai, Y. Liu, and M. Xie, "A dynamic-Bayesian-network-based fault diagnosis methodology considering transient and intermittent faults," *IEEE Trans. Automat. Sci. Eng.*, vol. 14, no. 1, pp. 276–285, Jan. 2017.
[44] M. Pecht and V. Ramappan, "Are components still the major problem: A review of electronic system and device field failure returns," *IEEE Trans. Compon., Hybrids, Manuf. Technol.*, vol. 15, no. 6, pp. 1160–1164, Dec. 1992.
[45] C. Wei, M. Benosman, and T. Kim, "Online parameter identification for state of power prediction of lithium-ion batteries in electric vehicles using extremum seeking," *Int. J Control Autom.*, vol. 17, pp. 2906–2916, Aug. 2019.
[46] Z. H. Gao, C. B. Ma, Y. G. Luo, and Z. Y. Liu, "IMA health state evaluation using deep feature learning with quantum neural network," *Eng. Appl. Artif. Intel.*, vol. 76, pp. 119–129, 2018.
[47] G. Li, P. Niu, X. Duan, and X. Zhang, "Fast learning network: A novel artificial neural network with a fast learning speed," *Neural Comput. Appl.*, vol. 24, pp. 1683–1695, Apr. 2013.
[48] M. H. Ali, B. A. D. A. Mohammed, A. Ismail, and M. F. Zolkipli, "A new intrusion detection system based on fast learning network and particle swarm optimization," *IEEE Access*, vol. 6, pp. 20255–20261, 2018.