Hindawi
Complexity
Volume 2018, Article ID 3813029, 13 pages
https://doi.org/10.1155/2018/3813029

Research Article

Predicting the Remaining Useful Life of an Aircraft Engine Using a Stacked Sparse Autoencoder with Multilayer Self-Learning

Jian Ma,[1,2] **Hua Su**,[1,2] **Wan-lin Zhao**,[1,2] **and Bin Liu**[1,2]

[1]*School of Reliability and Systems Engineering, Beihang University, Beijing, China*
[2]*Science & Technology on Reliability and Environmental Engineering Laboratory, Beijing, China*

Correspondence should be addressed to Bin Liu; liubin6862@126.com

Received 6 January 2018; Revised 22 April 2018; Accepted 3 May 2018; Published 30 July 2018

Academic Editor: Minvydas Ragulskis

Because they are key components of aircraft, improving the safety, reliability and economy of engines is crucial. To ensure flight safety and reduce the cost of maintenance during aircraft engine operation, a prognostics and health management system that focuses on fault diagnosis, health assessment, and life prediction is introduced to solve the problems. Predicting the remaining useful life (RUL) is the most important information for making decisions about aircraft engine operation and maintenance, and it relies largely on the selection of performance degradation features. The choice of such features is highly significant, but there are some weaknesses in the current algorithm for RUL prediction, notably, the inability to obtain tendencies from the data. Especially with aircraft engines, extracting useful degradation features from multisensor data with complex correlations is a key technical problem that has hindered the implementation of degradation assessment. To solve these problems, deep learning has been proposed in recent years to exploit multiple layers of nonlinear information processing for unsupervised self-learning of features. This paper presents a deep learning approach to predict the RUL of an aircraft engine based on a stacked sparse autoencoder and logistic regression. The stacked sparse autoencoder is used to automatically extract performance degradation features from multiple sensors on the aircraft engine and to fuse multiple features through multilayer self-learning. Logistic regression is used to predict the remaining useful life. However, the hyperparameters of the deep learning, which significantly impact the feature extraction and prediction performance, are determined based on expert experience in most cases. The grid search method is introduced in this paper to optimize the hyperparameters of the proposed aircraft engine RUL prediction model. An application of this method of predicting the RUL of an aircraft engine with a benchmark dataset is employed to demonstrate the effectiveness of the proposed approach.

1. Introduction

Because they are core components of an aircraft, the failure of engines is often a major cause of major accidents and casualties [1]. Therefore, the safety and the reliability of engines are vital to the performance of aircraft. However, it is difficult to ensure their safety and reliability due to their complicated structures, and engine failure has arisen inevitably due to effects of aging, environment, and variable loading as the working time increases. For this reason, it is essential to detect underlying degradation, predict how soon an engine will fail effectively, implement maintenance promptly, and ultimately prevent catastrophic failure.

In the field of aircraft maintenance, traditional maintenance is either purely reactive (fixing or replacing an aircraft engine component after it fails) or blindly proactive (assuming a certain level of performance degradation with no input from the aircraft engine itself and maintaining the aircraft engine on a routine schedule whether maintenance is actually needed or not). Both scenarios are quite wasteful and inefficient, and neither is conducted in real time [2–5]. Given the scheduling of maintenance tasks based on fault diagnosis, performance degradation assessment and the predicted remaining useful life of the aircraft equipment and the need to prevent faults in advance, prognostics and health management (PHM) is gradually replacing these two maintenance

strategies. Prognostics, as the core of PHM, involves managing performance deterioration processes or faults in the aircraft engine and forecasts when components/systems of the engine will breakdown or when the performance will reach to an unacceptable level.

There are three main classes of RUL prediction methods: (1) data-driven methods, (2) physics model-based methods, and (3) methods that combine data-driven and physics model-based methods [6–9]. The data-driven methods use past condition monitoring data, the current health status of the system, and data on the degradation of similar systems. The methods based on physics models use system-specific mechanistic knowledge, failure regulation, and condition monitoring data to predict the RUL of a system or component. There are two main challenges in prognostics based on physics: (1) there is not enough physical knowledge to construct a physical degradation model and (2) the values of the physical model's parameters are difficult to determine exactly. Therefore, it is important to understand the failure mechanism of the system correctly, and experienced personnel are required for physics-based models [10, 11]. In addition, the peripheral environment during device operation (e.g., the temperature and humidity) and the operating conditions (e.g., the fan speed) may be used as inputs and constitute additional dimensions to be considered. Therefore, the requirements of data-driven methods to model the degradation and predict the RUL are easier to satisfy in reality. At present, data-driven methods are widely used in RUL prediction [12, 13].

The performance of many data-driven prognostics methods is heavily dependent on the choice of the performance degradation data to which they are applied [14]. However, engines have many sensor parameters. The sensitivity of the data from different sensors varies in terms of showing engine performance degradation; the data from some sensors is sensitive and the data from other sensors is not sensitive. Therefore, it is necessary to select suitable sensor parameters whose data are more sensitive to the engine's performance degradation trend as the training data for the RUL prediction model. By observing the characteristic variations of the data from all sensor parameters, quadratic fitting curve is used to fit the degradation data from different sensors and rank the engine's sensor parameters by sensitivity.

Three problems hinder the implementation of performance degradation feature extraction in practice. The first is to select the most sensitive performance degradation features for identifying performance degradation trends easily. The second is that the relevant performance degradation features are often not available and unknown a priori; a large number of candidate performance degradation features have been proposed to better represent the performance degradation state. The last is that most traditional methods of extracting performance degradation features for prognostics are unsupervised and cannot automatically adjust the feature extraction modal parameters based on feedback from the prediction [15–17]. Such feature extraction and choice is significant but represents a principal shortcoming of popular prognostics algorithms: the inability to extract and organize discriminative or trend information from data. Therefore, it is important to develop an automatic feature extraction method that is capable of extracting the prominent feature to achieve better insight into the underlying performance degradation state.

Deep learning, a new method that has been put forward in the last few years, can be used to extract multilevel features from data, which means the method could express data at different levels of abstraction [18]. Deep learning is an end-to-end machine learning system. It can automatically process an original signal, identify discriminative and trend feature in the input data layer by layer, and then, directly output the classification/regression result. The whole process of feature learning and classifier/regression model training is based on optimizing an overall objective function. In contrast, traditional machine learning processes are divided into several discontinuous data preprocessing steps, such as manual feature extraction and classifier/regression model training, and each step is based on optimizing a separate objective function. Due to the advantage of feature self-learning, deep learning has had great success in applications in artificial intelligence, including computer vision (CS), natural language processing (NLP) [19, 20], object recognition [21], and image information retrieval [22, 23]. Deep learning is not only popular in the academic world but also favored in the industrial world. Companies such as Google, Microsoft, Apple, IBM, and Baidu [24], whose products are widely used, are researching deep learning and have made achievements, such as AlphaGo.

There are many deep learning methods: deep neural networks (DNNs), convolutional deep neural networks (CNNs), deep belief networks (DBNs), and so on [25], for instance, have been proposed. The stacked sparse autoencoder (SAE) [26] is one of the most commonly used deep neural network approaches. SAE consists of multilayer autoencoder such as sparse autoencoder, denoising autoencoder, and so on. Sparse autoencoder is on the basis of autoencoder and introduced sparse constraint condition to aid the expression code as sparse as possible. Denoising autoencoder can learn to remove the noise which is added to the initial input data and extract more robust expression of the input data [27]. For this reason, SAE can effectively capture the important factor of input data, extract more helpful and robust features of data, and then realize excellent performance in pattern recognition and machine learning.

In recent years, various researchers have demonstrated the success of DNN and SAE models in the application of machine health monitoring, such as fault classification of induction motor operated under six different conditions, vibration based fault diagnosis of rolling bearing and hydraulic pump, fault detection within tidal turbine's generator from vibration data acquired from an accelerometer sensor placed within the nacelle of the turbine, vibration based condition monitoring of air compressors, multi class fault classification of spacecraft using large variety of data generated during the spacecraft test, anomaly detection and fault disambiguation in large flight data, drill bit and steel plate health monitoring using vibration data, fault recognition of voltage transformer in electric power industry and so on [28–36]. Most of the research of SAE based

health monitoring mainly focus on anomaly detection and fault diagnosis at present. However, there are few applications on RUL prediction, especially for aircraft engine RUL prediction.

Consequently, a prognostics method based on a stacked sparse autoencoder is proposed to promote self-learning of multilayer features and to predict the RUL of an aircraft engine. The remainder of this paper is organized as follows: Section 2 presents the entire prediction method procedure and framework. Section 3 presents and discusses the prediction results. Finally, conclusions are drawn in Section 4.

2. Methodology

This section introduces the relevant algorithms used in this research. As depicted in Figure 1, the whole procedure for RUL prediction for an aircraft engine consists of two main steps: data preprocessing and RUL prediction using the SAE.

2.1. Data Preprocessing. Selection of sensors that are sensitive to performance degradation and standardization of sensor data with different dimensions are the primary tasks necessary to obtain a high RUL prediction accuracy. Three steps are needed to preprocess the data.

2.1.1. Sensor Selection. Different sensors in an aircraft engine have very different responses to the performance degradation process. Some sensors show unclear tendencies because of noise or insensitivity to degradation trends. Choosing insensitive parameter data may reduce the RUL prediction accuracy. To improve the performance of the prediction model, sensors that are more sensitive to the performance degradation process are chosen as inputs to the RUL prediction model. A method called slope analysis is proposed for sensitivity measurement. Its three main steps are as follows:

Step 1: curve fitting is performed on the degradation data for each parameter of each engine. Then, the parameters of the best-fit curves, called slopes, are used to analyze the sensitivity of the degradation data.

Step 2: the average values of all the engine parameters in the step 1 that belong to the same sensor are calculated. Then, the different average parameter values for the different sensors show the individual sensitivity of the degradation data.

Step 3: the degradation data with larger slopes are selected for predicting the RUL of the engine.

2.1.2. Data Normalization. The linear function that best preserves the original performance degradation pattern of the aircraft engine is chosen to map the data for each selected sensor to [0, 1].

2.1.3. RUL Normalization. The proposed prediction method outputs a result in the range from 0 to 1. In the training stage of the prediction model, the RUL of each cycle of aircraft engine should also be normalized to [0, 1] using a linear function. The test outputs of the prediction model need to be inversely mapped from [0, 1] to the real RUL.

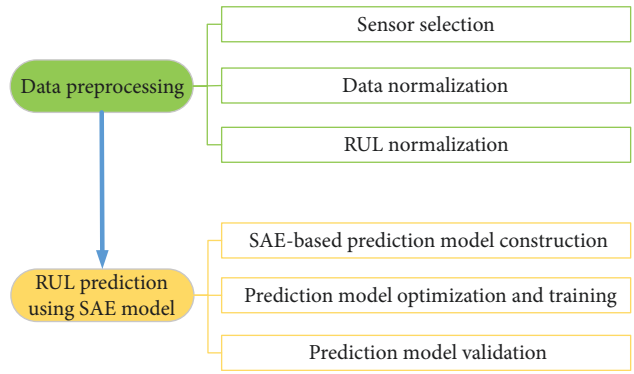

Figure 1: The procedure for predicting the RUL of an aircraft engine.

2.2. SAE Model Construction

2.2.1. Deep Architecture. Cortical computations in the brain have deep architecture and multiple layers of processing. For example, a visual image is processed in multiple stages by the brain, first by cortical area "V1," then by cortical area "V2," and so on [37]. Inspired by the information-processing scheme of the brain, deep neural networks have similar deep architectures and multiple hidden layers, which can support complex recognition tasks [6, 37]. As is typical of deep neural networks, the stacked sparse autoencoder (SAE) consists of multiple autoencoders. Compared with traditional neural networks with shallow architectures, it can learn features better and extract deeper discriminative representations [38].

However, it is difficult to train deep architectures [39]. This problem has been addressed by Hinton et al. [40–42], who showed that deep architectures can be trained by relying on two main procedures: (1) on the basis of a unsupervised autoencoder, the deep architecture layers are processed by pretraining, and the output of the top layer's autoencoder is used as the input to a logistic regression and (2) fine-tuning based on backpropagation is used to adjust the model parameters to obtain accurate prediction results.

2.2.2. Sparse Autoencoder. An autoencoder, first introduced by Hinton et al. [40], is a general form of deep learning method [43] that has been extensively used in unsupervised feature learning. As shown in Figure 2, an autoencoder has three layers: an input layer, a hidden layer, and an output layer. The whole network is trained to realize the reconstruction from the input layer to the output layer, while the hidden layer is accepted as the key feature. However, the traditional autoencoder is not an efficient way to obtain significant representativeness due to its intrinsic limitations. The SAE, as an extension of an autoencoder, can be trained to obtain relatively sparse representatives by introducing a sparse penalty term into the autoencoder [44]. The sparse features learned by the SAE have meanings that are more practical in experiments and applications.

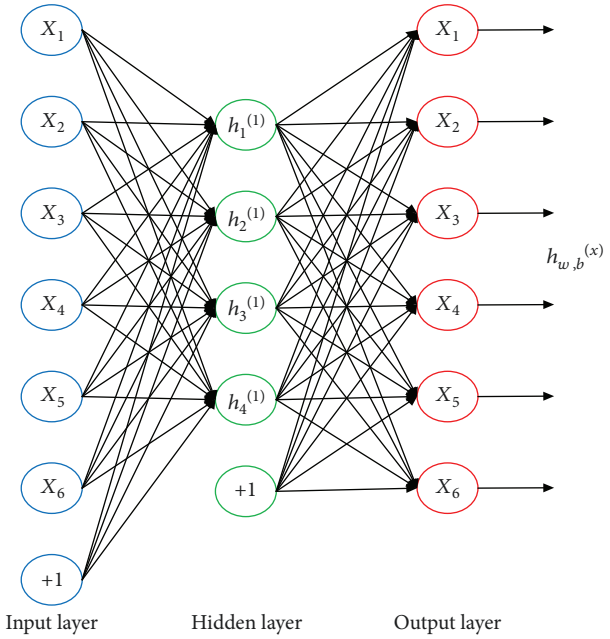

Figure 2: The structure of an autoencoder.

The SAE model contains two parts:

(i) An encoder map

The encoder maps an input vector $x^{(i)} \in \mathrm{R}^{d_0}$ (the ith training example) to the latent representation $a^{(i)}$ through deterministic mapping

$$a^{(i)} = f_1\left(x^{(i)}\right) = \text{sigmoid}\left(W_1 x^{(i)} + b_1\right), \quad (1)$$

where sigmoid is the activation function of the encoder with weight matrix W_1 and *bias vector* b_1.

(ii) A decoder map

The decoder maps feature $a^{(i)}$ back to a reconstruction of the vector $h_{W,b}(a^{(i)}) \in R^{d_0}$ in the output space [45] through a mapping function

$$\widehat{x}^{(i)} = h_{W,b}\left(a^{(i)}\right) = \text{sigmoid}\left(W_{11} a^{(i)} + b_2\right). \quad (2)$$

The decoder map tries to learn a function $h_{W,b}(a^{(i)}) \approx x^{(i)}$, which means making the output $\widehat{x}^{(i)}$ similar to the input $\widehat{x}^{(i)}$. Similarly, sigmoid is set as the activation function of the decoder map with weight matrix W_{11} and bias vector b_2.

During the learning process, the parameters of the SAE are adjusted using backpropagation by minimizing the cost function within the sparsity constraint. The sparsity constraint works on the hidden layer to limit its units and makes it into a sparse vector in which most elements are zero or close to zero [44]. For the autoencoder's network structure, a neuron with a sigmoid activation function is in the active state if its output is close to 1 and the inactive state if its output is close to 0. Therefore, the sparsity constraint is introduced to restrict most of the neurons to inactivity most of the time.

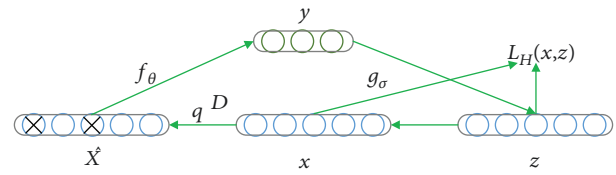

Figure 3: The concept of the denoising autoencoder.

The activation of hidden unit j is denoted by $a_j(x)$, and the average activation of hidden unit j is as follows:

$$\widehat{\rho}_{\mathrm{j}} = \frac{1}{\mathrm{m}} \sum_{i=1}^{\mathrm{m}} \left[a_j^{(\mathrm{i})}\left(x^{(i)}\right)\right]. \quad (3)$$

Then, we define the sparsity constraint as $\widehat{\rho}_{\mathrm{j}} = \rho$, where ρ denotes the sparsity criterion and has a value that is close to zero, that is, most of the neurons in the hidden layer are inactive.

To reach the goal of sparsity, a penalty term is introduced to the objective function that penalizes $\widehat{\rho}_{\mathrm{j}}$ if it deviates significantly from ρ. In our study, the KL divergence [45] is selected as the penalty term;

$$\mathrm{KL}\left(\rho \middle\| \widehat{\rho}_j\right) = \rho \log \frac{\rho}{\widehat{\rho}_j} + (1-\rho) \log \frac{1-\rho}{1-\widehat{\rho}_j}. \quad (4)$$

The training set of m training examples is denoted by $\{(x^{(1)}, y^{(1)}) \ldots, (x^{(m)}, y^{(m)})\}$, and the original cost function is defined as

$$\begin{aligned} J(W,b) &= \left[\frac{1}{m}\sum_{i=1}^{m} J\left(W,b;x^{(i)};y^{(i)}\right)\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ji}^{(l)}\right)^2 \\ &= \left[\frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}\left\|h_{W,b}\left(\mathrm{x}^{(i)}\right) - y^{(i)}\right\|^2\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ji}^{(l)}\right)^2. \end{aligned} \quad (5)$$

The first term in (5) is an average sum-of-squares error term, and the second term is a regularization term or weight decay term, which tends to decrease the magnitude of the weights. Here, W and b are the same as in (1) and (2), and λ is the weight decay parameter.

By adding the sparse penalty term, the cost function is modified to

$$\mathrm{J}(W,b)_{\text{sparse}} = J(W,\mathrm{b}) + \beta \sum_{j=1}^{s_2} KL\left(\rho \middle\| \widehat{\rho}_j\right), \quad (6)$$

where β represents the weight of the sparsity penalty term.

2.2.3. Denoising Autoencoder. Despite the process described above, learning features well to improve the performance and generalization ability of the prediction model continues to face challenges because of the noise and outliers that commonly appear in real-world data. To force the hidden layer to

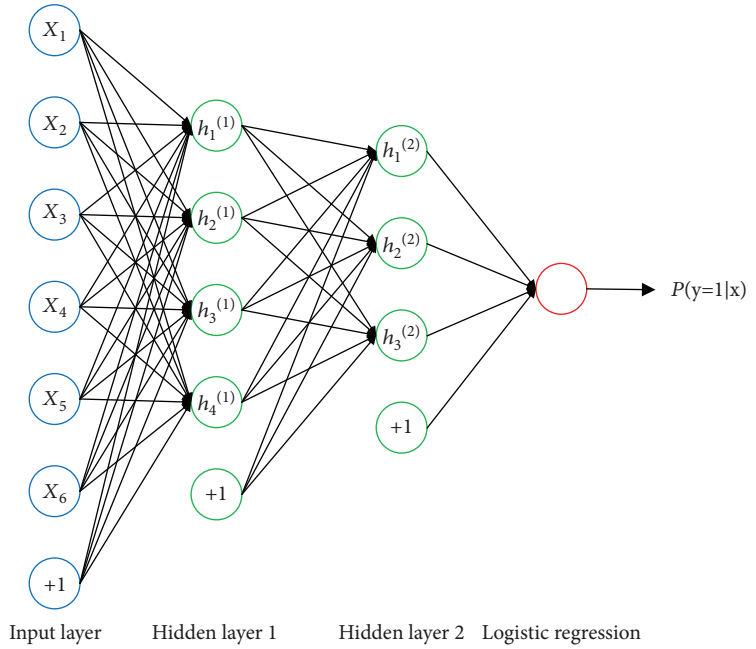

Figure 4: A two-layer stacked sparse autoencoder and logistic regression.

discover more robust features, the autoencoder can be trained by reconstructing the input from a corrupted version of it, which is the idea behind denoising autoencoders [37], as shown in Figure 3.

These data corruption is implemented by corrupting the initial input x to create a partially destroyed version $\tilde{x}$ by means of a stochastic mapping,

$$\tilde{x} \sim q_D(\tilde{x}|x). \tag{7}$$

The standard approach is to apply masking noise to the original data by setting a random fraction of the elements of x to zero. Next, the corrupted data $\tilde{x}$ pass through a basic autoencoder process and is mapped to a hidden representation,

$$y = f_\theta(\tilde{x}) = \text{sigmoid}(W.\tilde{x} + b). \tag{8}$$

From this equation, we reconstruct

$$z = g_{\theta'}(y). \tag{9}$$

In the last stage, the parameters are trained to minimize the average reconstruction error

$$L_{\text{H}}(x, z) = \text{H}(\text{B}_x \| \text{B}_z) \tag{10}$$

to make z as close as possible to the uncorrupted input x.

2.2.4. Structure of the Stacked Sparse Autoencoder. As a typical neural network, the stacked autoencoder consists of multiple layers of sparse or denoising autoencoders (discarding the decoder) and a logistic regression. The outputs of each layer of the stacked autoencoder are wired to the inputs of the subsequent layer. The architecture of a two-layer stacked sparse autoencoder is shown in Figure 4. Each sparse or denoising autoencoder generates a representation of the inputs (data from the aircraft engine's sensors) that is more abstract and high dimensional than the previous layer's because it is obtained by applying an additional nonlinear transformation. The output of the last layer of the sparse autoencoder are input to the logistic regression and then, the results (the predicted RUL) are obtained.

(1) Prediction Using Logistic Regression. The purpose of logistic regression is to find an optimal model for matching independent variables and class distinctions of dependent variables (probabilities of the occurrence of an event). The logistic function is expressed by

$$\text{prob}(\text{event}) = p(x) = \left(1 + e^{-e(x)}\right)^{-1}. \tag{11}$$

The logistic or logit model is

$$\begin{aligned} \text{Logit} = g(x) &= \log\left(p(x)(1 - p(x))^{-1}\right) \\ &= \alpha + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k, \end{aligned} \tag{12}$$

where $g(x)$ is a linear combination of the independent variables $x_1 x_2, \ldots, x_k$.

Parameters of models (such as $\alpha, \beta_1, \ldots, \beta_k$) need to be determined beforehand, which is the major premise for determining $P(x)$. Because of the existence of dichotomous dependent variables, it is improper to estimate the values of the parameters using the least-squares method [46]. Therefore, compared with the method of minimizing the sum of the squared errors, the paper uses the maximum likelihood method to estimate the parameters (such as $\alpha, \beta_1, \ldots, \beta_k$) of the logistic regression [47]. Then, the probability of the occurrence of the event can be obtained using (11) once the vector x has been determined.

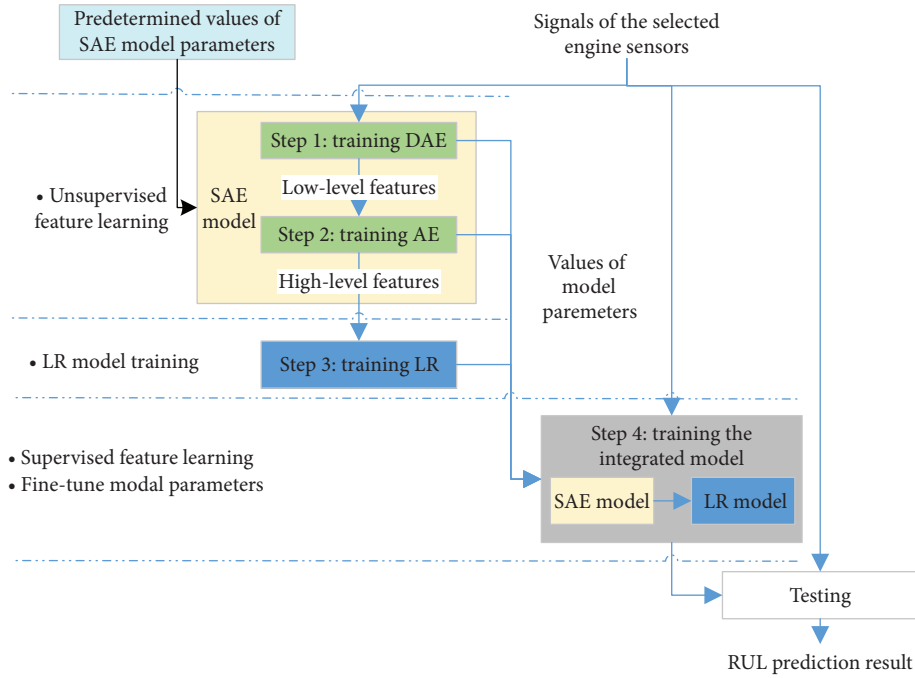

Figure 5: Procedure for training the SAE-based RUL prediction model.

(2) Fine Tuning. The process of fine-tuning mainly focuses on adjusting the weights in the SAE network, which leads to much better prediction performance.

First, feed-forward is used to compute the activations for all the autoencoder layers.

In the next step, we set $\delta^{(n_l)} = -(\nabla_{a^{n_l}} J) \bullet f'(z^{n_l})$ for the output layer, where $\nabla J = \theta(I - P)$, I is the input label, and P is the vector of conditional probabilities. Then, for layers $l = n_l - 1, n_l - 2, \ldots, 3, 2$, we set $\delta^{(l)} = ((W^{(l)})^T \delta^{(l+1)}) \bullet f'(z^{n_l})$, and then, the desired partial derivatives are

$$\begin{aligned} \nabla W^l J(W, b; x; y) &= \delta^{(l+1)} \left(a^{(l)}\right)^T, \\ \nabla b^l J(\mathrm{W}, \mathrm{b}; \mathrm{x}; \mathrm{y}) &= \delta^{(l+1)}, \end{aligned} \tag{13}$$

where W, b, and a are as in (1) and (2).

Finally, the batch gradient descent algorithm is used to minimize the overall cost function.

2.3. Training and Optimization of SAE-Based RUL Prediction

2.3.1. Procedure for Training the SAE-Based RUL Prediction Model. A two-layer stacked sparse autoencoder and a logistic regression (LR) model were used as an example to illustrate the training procedures in the proposed deep learning-based RUL prediction methodology. The values of the SAE parameters are predetermined. A grid search is used to find a set of optimal SAE parameters. The four major steps of the procedure are as follows:

Step 1: a single-layer denoising autoencoder (DAE), the first layer of the SAE, is trained to extract robust performance degradation features using unsupervised learning [37]. The signals of the selected sensors are input into the DAE, and then, low-level features are output by the hidden layer of the DAE.

Step 2: a single-layer sparse autoencoder (AE), the second layer of the SAE, is trained for unsupervised self-learning of features. The low-level features are input into the AE, and the high-level features are output by the hidden layer of the AE.

Step 3: high-level features are used as inputs to train the LR model for RUL prediction. The target output of the LR model is the normalized RUL of the aircraft engine.

Step 4: the previously trained SAE and LR model are combined into an integrated feature learning and RUL prediction model. Then, the integrated model is trained using supervised learning for the final feature learning to obtain the RUL

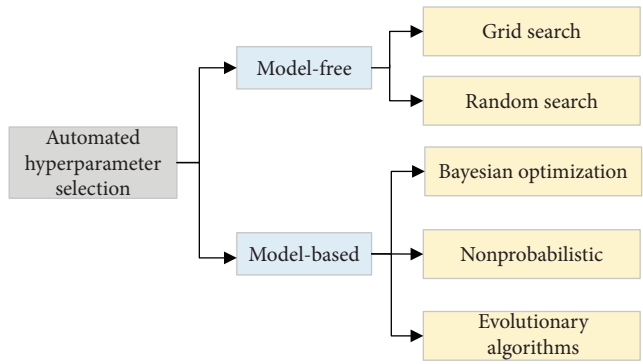

Figure 6: Approaches to automated hyperparameter selection.

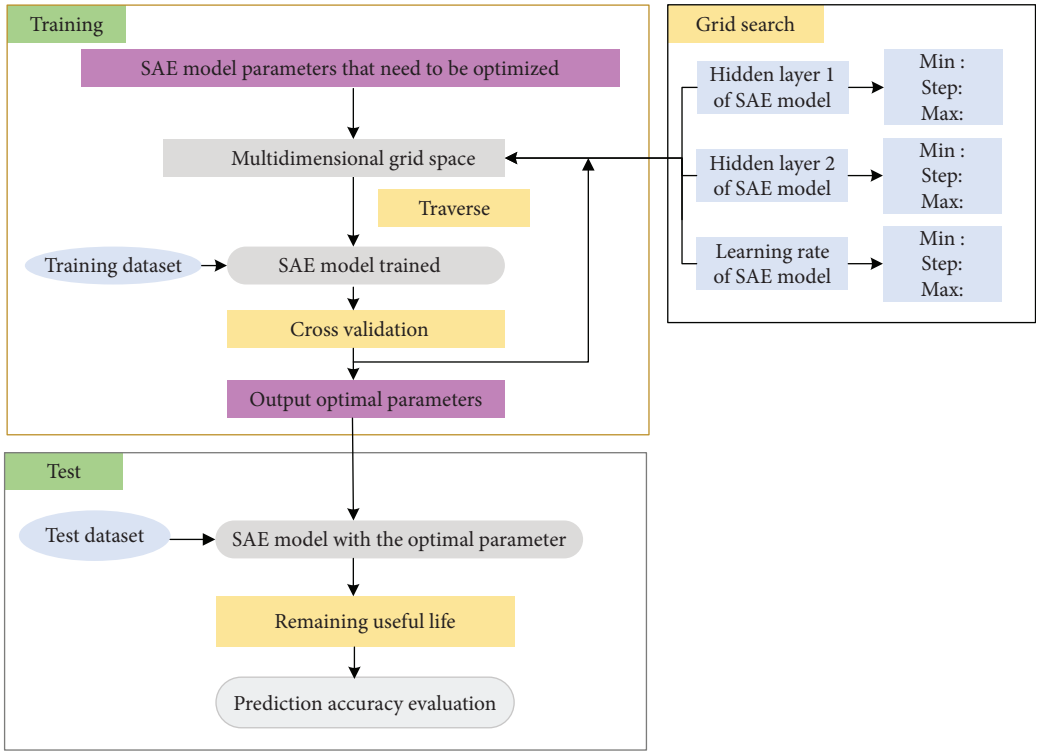

Figure 7: Grid search-based SAE RUL prediction model parameter optimization and validation.

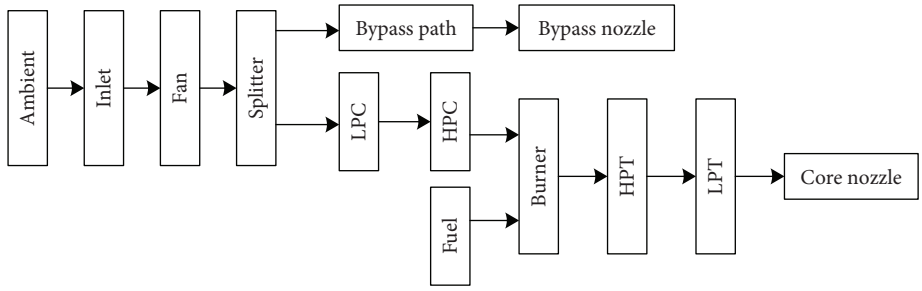

Figure 8: Connections of various modules in the simulation environment.

prediction model. The signals of the selected engine sensors are the inputs of the integrated model, and the normalized RUL of the engine is used as the target output during model training. Training the integrated model using supervised learning can fine-tune the modal parameters (the parameters of the DAE, the AE, and the LR models) based on the values obtained in the previous training, steps 1 to 3. The features obtained from the fine-tuned DAE and AE more clearly present the degradation trend of the engine performance. Based on these features, the LR model can provide a more accurate RUL prediction result.

The process of training the proposed RUL prediction method is summarized in Figure 5.

2.3.2. Grid Search-Based SAE RUL Prediction Model Parameter Optimization and Validation. The hyperparameters of the deep learning, which have significant impacts on the feature extraction performance, are adjusted in most cases based on expert experience. In view of the difficulty of adjusting hyperparameters by means of deep learning, a method of optimizing the hyperparameters is necessary.

There are currently two main types of method of automated hyperparameter selection for the SAE (which is shown in Figure 6). One includes model-free methods, which include the grid and random search methods; the other includes model-based methods, which mainly include three subcategories, the Bayesian optimization (e.g., spearmint [48]), nonprobabilistic methods (such as RBF surrogate models [49]), and evolutionary algorithms (e.g., genetic algorithms [50] and particle swarm optimization [51]). Model-based methods efficiently explore the solution space

according to the algorithm selected and then, quickly obtain the accepted parameter value. However, the identified hyperparameter value may be a local optimum, and the method has several individual hyperparameters, which would increase its complexity.

Unlike model-based methods of hyperparameter selection, model-free hyperparameter selection methods search for the optimal parameters within the defined space; the main ones are grid and random searching [52, 53]. In the paper, the grid search method is chosen to search for the hyperparameters of the SAE.

There are a few reasons why grid search is chosen as the hyperparameter optimization algorithm used in the proposed SAE-based RUL prediction model.

(1) Compared with the manual search method of optimizing the hyperparameters, a grid search is more likely to identify better model parameters than pure manual sequential optimization (in the same time).

(2) Compared with model-based hyperparameter selection methods, a grid search is simple to implement, and parallel computing is easy to implement.

(3) Compared with the random search method of hyperparameter optimization, mesh searches are recommended when few parameters need to be optimized.

Theoretically, when the space defined by the optimized parameters is large enough and the changes in the optimal parameters are small enough, the optimization method called mesh searching could be used to find the global optimal solution.

There are three main steps in the grid search-based hyperparameter optimization of the SAE RUL prediction model.

Step 1: the hyperparameters to be optimized are defined in the space, and the space is divided into grids with a fixed step size. Each point on each grid is a combination of model parameters.

Step 2: the training set is divided into several subsets of equal size. Then, the SAE is trained with one combination of model parameters. Details of the procedure for training the SAE-based RUL prediction model are in Section 2.3.1 of this paper.

Step 3: step 2 is repeated until the grid search has been completed. The resulting optimal hyperparameters are output.

Figure 7 shows the process of the method. To obtain an SAE with the optimal parameters, predicted degradation data for the engines are input into the trained model, and the RUL of each engine is obtained.

3. Case Study

3.1. Engine Data Description. The challenge datasets used for the prognostics challenge competition at the 2008 PHM International Conference consist of multiple multivariate time series, which were collected via a dynamical simulation of an engine system. The model simulated various degradation scenarios in any of the five rotating components of the simulated engine (fan, LPC, HPC, HPT, and LPT), and the connections among the engine modules in the simulation are shown in Figure 8. The engine begins in normal operation, then, degradation appears in some cycle of the simulation. The degradation data for each engine are recorded until the engine fails. The simulation model results in 218 engine datasets defined as unit 1 through unit 218 with different failure times measured by the number of operating cycles for the same engine system.

The complete dataset for each cycle of each engine unit consists of the unit ID, the operating cycle index, the operational regime settings, and typical sensor measurements. A total of 21 sensors (shown in Table 1) are installed in different components of the aircraft engine. A total of 21 sensory signals are obtained under the R1 operation regime shown in Table 2. In this study, sensor data were collected from 200 aircraft engines injected with the HPC degradation

Table 1: Description of the sensor signals for the aircraft gas turbine engine.

Index	Symbol	Description	Unit
1	T2	Total temperature at fan inlet	°R
2	T24	Total temperature at LPC outlet	°R
3	T30	Total temperature at HPC outlet	°R
4	T50	Total temperature at LPT outlet	°R
5	P2	Pressure at fan inlet	psia
6	P15	Total pressure in bypass-duct	psia
7	P30	Total pressure at HPC outlet	psia
8	Nf	Physical fan speed	rpm
9	Nc	Physical core speed	rpm
10	epr	Engine pressure ratio (P50/P2)	—
11	Ps30	Static pressure at HPC outlet	psia
12	phi	Ratio of fuel flow to Ps30	pps/psi
13	NRf	Corrected fan speed	rpm
14	NRc	Corrected core speed	rpm
15	BPR	Bypass ratio	—
16	farB	Burner fuel-air ratio	—
17	htBleed	Bleed enthalpy	—
18	Nf_dmd	Demanded fan speed	rpm
19	PCNFR_dmd	Demanded corrected fan speed	rpm
20	W31	HPT coolant bleed	lbm/s
21	W32	LPT coolant bleed	lbm/s

°*R*: Rankine temperature scale; psia: pounds per square inch absolute; rpm: revolutions per minute; pps: pulses per square inch; lbm/s: pound mass per second.

Table 2: Operating regime of the aircraft engine.

Operation regime	Altitude	Mach number	Throttle resolver angle (TRA)
R1	20	0.7	0

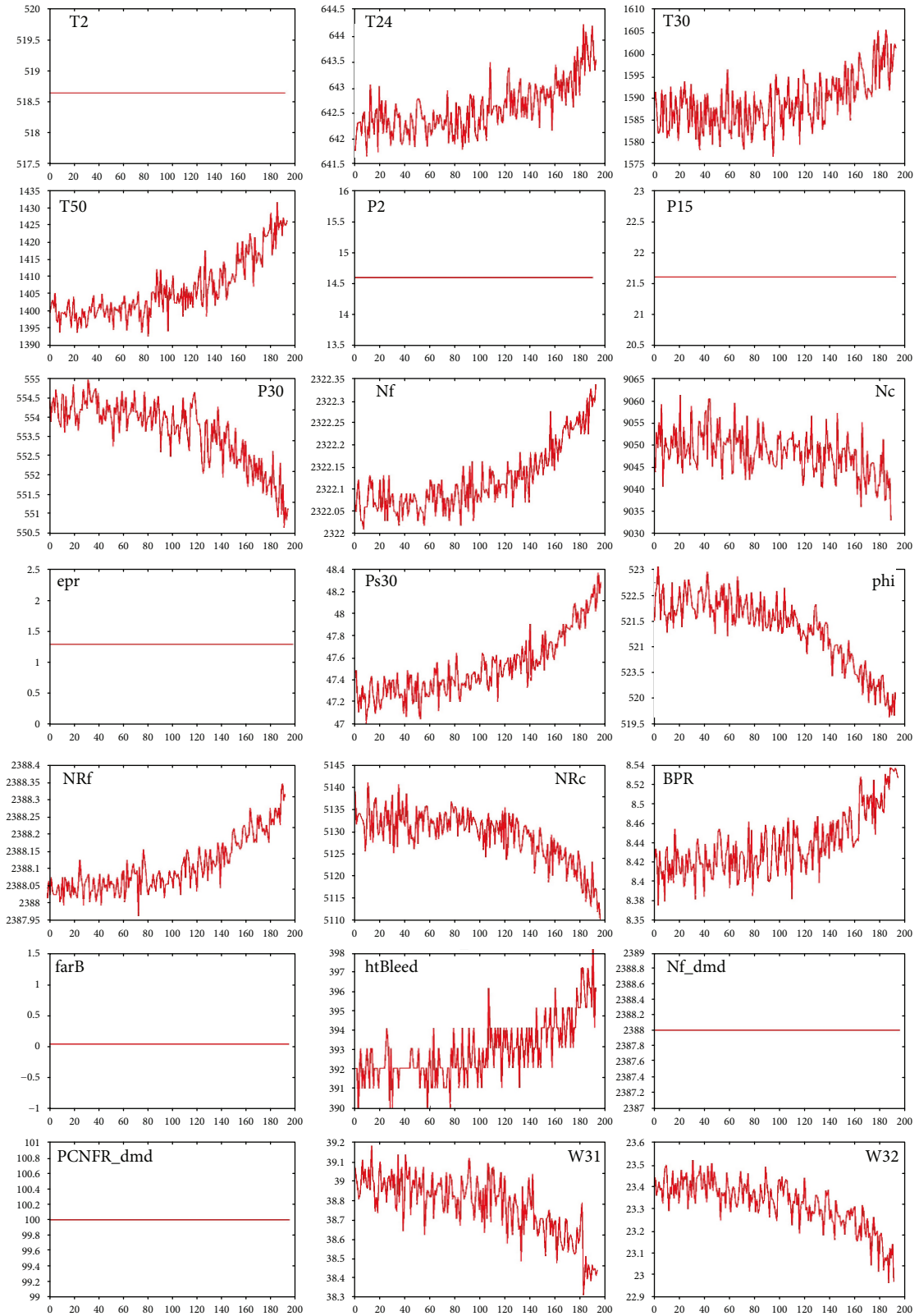

Figure 9: Engine degradation tendencies using the data from different sensors over time.

fault mode. The dataset considered in this study consists of three files, which include degradation data for 100 training and 100 testing units and the remaining useful life of the 100 testing units. Each training unit runs to breakdown, and each testing unit stops running at some time before it breaks down. Through investigation and research,

Table 3: Operation regime of the aircraft engine.

Rank	Sensor	Sensitivity
1	Ps30	2.63785E-05
2	phi	2.58711E-05
3	P30	2.45237E-05
4	T50	2.43431E-05
5	BPR	2.35888E-05
6	Nf	2.30928E-05
7	NRf	2.25432E-05
8	W32	2.17608E-05
9	W31	2.1399E-05
10	htBleed	2.1267E-05
11	T24	2.1055E-05
12	T30	1.9856E-05
13	Nc	1.1581E-05
14	NRc	5.86587E-06

Table 4: Parameters of the SAE obtained using the grid search method.

Parameter name	Parameter value
Input layer	16
Learning rate of NN model	0.5
Number of training cycles	300

it is found that the dataset is highly authoritative and accurate [54–56].

3.2. Results and Discussion

3.2.1. Data Preprocessing. In the process of engine performance degradation, the performance data from the sensors gradually change over time, and the data indirectly reflect the degradation tendency of the engine's performance. However, the sensitivity of different parameters to degradation varies over time. Figure 9 shows the degradation tendency of the 21 performance parameters. According to Figure 9, data from seven of the sensors (1, 5, 6, 10, 16, 18, and 19) exhibit no tendency, so the sensitivities of the remaining parameters to engine performance degradation are analyzed. The results of the 14 performance parameters for which the sensitivity analysis is conducted are shown in Table 3. To reduce the computational complexity, the data from the first six sensors (4, 7, 8, 11, 12, and 15) in the sensitivity ranking are selected. By surveying and analyzing relevant information about the RUL of engines, parameters T24 and T30 are also chosen as objects of study. Finally, eight performance parameters (2, 3, 4, 7, 8, 11, 12, and 15) are chosen for predicting the RUL of the aircraft engine [57–59].

3.2.2. SAE Parameters Optimized Using Grid Searching. The SAE used in this paper has eight hyperparameters: input layer, hidden layer 1, hidden layer 2, output layer, learning rate of SAE, learning rate of NN model, and number of training cycles. Based on the results, the parameters input layer, learning rate of NN model, and number of training cycles are shown in Table 4, which is a good parameter match. Then, there are three hyperparameters that need to be optimized using grid searching. The grid search method of automated hyperparameter selection for SAE is performed in a defined space with a fixed step size. The proposed parameters of the SAE obtained using the grid search method are shown in Table 5.

Table 5: Parameters of the SAE model obtained using the method.

Parameter name	Min	Step	Max
Hidden layer 1 of SAE	4	2	16
Hidden layer 2 of SAE	2	2	8
Learning rate of SAE	0.3	0.1	1

Table 6: Life prediction results for the first seven optimal parameters.

Rank	Accuracy rate	MAPE	Acceptable number	Acceptable rate
1	83.82%	16.18%	82	86.32%
2	83.67%	16.33%	81	85.26%
3	83.67%	16.33%	80	84.21%
4	83.66%	16.34%	80	84.21%
5	83.65%	16.35%	80	84.21%
6	83.62%	16.38%	81	85.26%
7	83.61%	16.39%	81	85.26%

Table 7: Engine life prediction accuracy of the 2008 PHM data challenge.

Rank	Score	MAPE	Accuracy rate
1	512.12	15.81%	84.19%
2	740.31	18.92%	81.08%
3	873.36	19.19%	80.81%
4	1218.43	20.15%	79.85%
5	1218.76	33.14%	66.86%
6	1232.27	32.90%	67.10%
7	1568.98	36.75%	63.25%
8	1645.77	30.00%	70.00%
9	1816.60	26.47%	73.53%
10	1839.06	27.72%	72.28%

3.2.3. Results. Through automated selection of the DNN hyperparameters using a grid search, the experimental results show that the method is effective, with an accuracy rate of up to 83.82% and an acceptable rate of up to 86.32% of ranking first (shown in Table 6). Compared with the accuracy of the 2008 PHM data challenge engine life prediction, the first-rank prediction accuracy is 84.19% (shown in Table 7), and the RUL prediction accuracy is quite close. However, there are six types of working condition and 218 training and test

Table 8: The first seven optimal parameter arrays obtained by the grid search method.

Rank	Input layer	Hidden layer 1	Hidden layer 2	Output layer	Learning rate of SAE model
1	16	10	2	1	0.9
2	16	10	2	1	0.5
3	16	8	2	1	0.3
4	16	8	2	1	0.5
5	16	12	2	1	0.9
6	16	10	2	1	0.7
7	16	12	4	1	0.5

sets in the FD005T dataset, which were used in the data challenge. Then, comparing the method proposed in this paper and the 2008 PHM data challenge provides only a relative comparison. The accuracy of the RUL predictions obtained in this paper is acceptable in the field of engine prediction, and the results are satisfactory. Table 8 shows the first seven optimal parameter arrays obtained by the grid search method. Table 6 shows the life prediction results based on the first seven optimal parameters. Table 7 shows the seven most accurate engine life predictions in the 2008 PHM data challenge.

4. Conclusions

In this paper, a new data-driven approach to engine prognostics is developed based on deep learning that can capture effective nonlinear features by themselves and reduce manual intervention. The SAE, a type of deep learning model, is not only able to capture the tendency of the system to evolve but also sufficiently robust to noise. To automatically select the hyperparameters of the SAE, the grid search algorithm is used. The method of predicting an aircraft engine's remaining useful life is applied to the 2008 PHM data challenge dataset to demonstrate the effectiveness of the proposed approach. The experimental results, which show a satisfactory prediction accuracy and acceptance rate for all the samples, show that the method is effective at predicting the RUL of an aircraft engine. It also has significance for enhancing the safety of aircraft engines and prognosticating and managing the health of aircraft engines to reduce the cost of maintenance.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (Grant nos. 51605014, 51105019, and 51575021), the Aviation Science Fund (Grant no. 20163351018), the Technology Foundation Program of National Defense (Grant no. Z132013B002), and the Fundamental Research Funds for the Central Universities (Grant no. YWF-18-BJ-Y-159).

References

[1] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *2008 International Conference on Prognostics and Health Management*, pp. 1–9, Denver, CO, USA, October 2008.

[2] S. H. Ding and S. Kamaruddin, "Maintenance policy optimization—literature review and directions," *International Journal of Advanced Manufacturing Technology*, vol. 76, no. 5–8, pp. 1263–1283, 2015.

[3] A. Bousdekis, B. Magoutas, D. Apostolou, and G. Mentzas, "Review, analysis and synthesis of prognostic-based decision support methods for condition based maintenance," *Journal of Intelligent Manufacturing*, vol. 29, no. 6, 2018.

[4] A. K. S. Jardine, D. Lin, and D. Banjevic, "A review on machinery diagnostics and prognostics implementing condition-based maintenance," *Mechanical Systems and Signal Processing*, vol. 20, no. 7, pp. 1483–1510, 2006.

[5] D. Djurdjanovic, J. Lee, and J. Ni, "Watchdog Agent—an infotronics-based prognostics approach for product performance degradation assessment and prediction," *Advanced Engineering Informatics*, vol. 17, no. 3-4, pp. 109–125, 2003.

[6] F. Ahmadzadeh and J. Lundberg, "Remaining useful life estimation: review," *International Journal of System Assurance Engineering and Management*, vol. 5, no. 4, pp. 461–474, 2014.

[7] A. Heng, S. Zhang, A. C. C. Tan, and J. Mathew, "Rotating machinery prognostics: state of the art, challenges and opportunities," *Mechanical Systems and Signal Processing*, vol. 23, no. 3, pp. 724–739, 2009.

[8] F. Camci and R. B. Chinnam, "Health-state estimation and prognostics in machining processes," *IEEE Transactions on Automation Science and Engineering*, vol. 7, no. 3, pp. 581–597, 2010.

[9] J. Luo, M. Namburu, K. Pattipati, L. Qiao, M. Kawamoto, and S. Chigusa, "Model-based prognostic techniques [maintenance applications]," in *Proceedings AUTOTESTCON 2003. IEEE Systems Readiness Technology Conference*, Anaheim, CA, USA, September 2003.

[10] O. F. Eker, F. Camci, and I. K. Jennions, "Major challenges in prognostics: study on benchmarking prognostics datasets," in *European conference of the prognostics and Health Management*, Dresden, Germany, 2012.

[11] H. Zhang, R. Kang, and M. Pecht, "A hybrid prognostics and health management approach for condition-based maintenance," in *2009 IEEE International Conference on Industrial Engineering and Engineering Management*, pp. 1165–1169, Hong Kong, China, December 2009.

[12] X.-S. Si, W. Wang, C.-H. Hu, and D.-H. Zhou, "Remaining useful life estimation – a review on the statistical data driven approaches," *European Journal of Operational Research*, vol. 213, no. 1, pp. 1–14, 2011.

[13] M. Schwabacher and G. Kai, "A survey of artificial intelligence for prognostics," Tech. Rep., AAAI Fall Symposium, 2007.

[14] N. Gebraeel, A. Elwany, and J. Pan, "Residual life predictions in the absence of prior degradation knowledge," *IEEE Transactions on Reliability*, vol. 58, no. 1, pp. 106–117, 2009.

[15] Y. Yang, Y. Liao, G. Meng, and J. Lee, "A hybrid feature selection scheme for unsupervised learning and its application in bearing fault diagnosis," *Expert Systems with Applications*, vol. 38, no. 9, pp. 11311–11320, 2011.

[16] 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," *Mechanical Systems and Signal Processing*, vol. 42, no. 1-2, pp. 314–334, 2014.

[17] R. F. Orsagh, J. Sheldon, and C. J. Klenke, "Prognostics/diagnostics for gas turbine engine bearings," in *2003 IEEE Aerospace Conference Proceedings (Cat. No.03TH8652)*, pp. 3095–3103, Big Sky, MT, USA, March 2003.

[18] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[19] O. Abdel-Hamid, L. Deng, D. Yu, and H. Jiang, "Deep segmental neural networks for speech recognition," *Interspeech*, vol. 36, p. 70, 2013.

[20] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, BC, Canada, March 2013.

[21] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, Columbus, OH, USA, June 2014.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, June 2016.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, pp. 1097–1105, Curran Associates, Inc., 2012.

[24] L. Deng and D. Yu, "Deep learning: methods and applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3-4, pp. 197–387, 2014.

[25] J. Schmidhuber, "Deep learning in neural networks: an overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[26] Y. Bengio and Y. LeCun, "Scaling learning algorithms towards AI," *Large-Scale Kernel Machines*, vol. 34, no. 5, pp. 1–41, 2007.

[27] S. Hu, Y. Zuo, L. Wang, and P. Liu, "A review about building hidden layer methods of deep learning," *Journal of Advances in Information Technology*, vol. 7, no. 1, pp. 13–22, 2016.

[28] W. Sun, S. Shao, R. Zhao, R. Yan, X. Zhang, and X. Chen, "A sparse auto-encoder-based deep neural network approach for induction motor faults classification," *Measurement*, vol. 89, pp. 171–178, 2016.

[29] T. Junbo, L. Weining, A. Juneng, and W. Xueqian, "Fault diagnosis method study in roller bearing based on wavelet transform and stacked auto-encoder," in *The 27th Chinese Control and Decision Conference (2015 CCDC)*, pp. 4608–4613, Qingdao, China, May 2015.

[30] Z. Huijie, R. Ting, W. Xinqing, Z. You, and F. Husheng, "Fault diagnosis of hydraulic pump based on stacked autoencoders," in *2015 12th IEEE International Conference on Electronic Measurement & Instruments (ICEMI)*, pp. 58–62, Qingdao, China, July 2015.

[31] G. S. Galloway, V. M. Catterson, T. Fay, A. Robb, and C. Love, "Diagnosis of tidal turbine vibration data through deep neural networks," in *Proceedings of the Third European Conference of the Prognostics and Health Management Society*, pp. 172–180, 2016.

[32] N. K. Verma, V. K. Gupta, M. Sharma, and R. K. Sevakula, "Intelligent condition based monitoring of rotating machines using sparse auto-encoders," in *2013 IEEE Conference on Prognostics and Health Management (PHM)*, pp. 1–7, Gaithersburg, MD, USA, June 2013.

[33] R. Thirukovalluru, S. Dixit, R. K. Sevakula, N. K. Verma, and A. Salour, "Generating feature sets for fault diagnosis using denoising stacked auto-encoder," in *2016 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pp. 1–7, Ottawa, ON, Canada, June 2016.

[34] R. Kishore, K. Reddy, S. Sarkar, and M. Giering, "Anomaly detection and fault disambiguation in large flight data: a multi-modal deep auto-encoder approach," in *Annual Conference of the Prognostics and Health Management Society*, PHM Society, Denver, Colorado, 2016.

[35] L. Wang, X. Zhao, J. Pei, and G. Tang, "Transformer fault diagnosis using continuous sparse autoencoder," *Springerplus*, vol. 5, no. 1, p. 448, 2016.

[36] K. Li and Q. Wang, "Study on signal recognition and diagnosis for spacecraft based on deep learning method," in *2015 Prognostics and System Health Management Conference (PHM)*, Beijing, China, October 2015.

[37] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning - ICML '08*, pp. 1096–1103, Helsinki, Finland, 2008.

[38] Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, vol. 7553, nature 521, 2015.

[39] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, Italy, March 2010.

[40] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[41] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, *Greedy Layer-Wise Training of Deep Networks*, Advances in neural information processing systems, 2007.

[42] Y. L. Boureau and Y. L. Cun, *Sparse Feature Learning for Deep Belief Networks*, Advances in neural information processing systems, 2008.

[43] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang, and S. Yan, "Sparse representation for computer vision and pattern recognition," in *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1031–1044, June 2010.

[44] N. Japkowicz, S. J. Hanson, and M. A. Gluck, "Nonlinear autoassociation is not equivalent to PCA," *Neural Computation*, vol. 12, no. 3, pp. 531–545, 2000.

[45] L. Chen, H. Qu, J. Zhao, B. Chen, and J. C. Principe, "Efficient and robust deep learning with correntropy-induced loss function," *Neural Computing and Applications*, vol. 27, no. 4, pp. 1019–1031, 2016.

[46] J. Yan and J. Lee, "Degradation assessment and fault modes classification using logistic regression," *Journal of Manufacturing Science and Engineering*, vol. 127, no. 4, pp. 912–914, 2005.

[47] W. M. Houston, D. J. Woodruff, and American College Testing Program, *Empirical Bayes Estimates of Parameters from the Logistic Regression Model*, ACT Research Report Series, ACT, Inc., 1997.

[48] J. Snoek, H. Larochelle, and R. P. Adams, *Practical Bayesian Optimization of Machine Learning Algorithms*, Advances in neural information processing systems, 2012.

[49] I. Ilievski, T. Akhtar, J. Feng, and C. A. Shoemaker, "Hyperparameter optimization of deep neural networks using non-probabilistic RBF surrogate model," *arXiv preprint arXiv*, vol. 1607, article 08316, 2016.

[50] K. Liu, L. M. Zhang, and Y. W. Sun, "Deep Boltzmann machines aided design based on genetic algorithms," *Applied Mechanics and Materials*, vol. 568–570, pp. 848–851, 2014.

[51] A. R. Syulistyo, D. M. J. Purnomo, M. F. Rachmadi, and A. Wibowo, "Particle swarm optimization (PSO) for training optimization on convolutional neural network (CNN)," *Jurnal Ilmu Komputer Dan Informasi*, vol. 9, no. 1, pp. 52–58, 2016.

[52] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. 1, pp. 281–305, 2012.

[53] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, *Algorithms for hyper-parameter optimization*, Advances in neural information processing systems, 2011.

[54] T. Wang, J. Yu, D. Siegel, and J. Lee, "A similarity-based prognostics approach for remaining useful life estimation of engineered systems," in *2008 International Conference on Prognostics and Health Management*, pp. 1–6, Denver, CO, USA, October 2008.

[55] J. Xu, Y. Wang, and L. Xu, "PHM-oriented integrated fusion prognostics for aircraft engines based on sensor data," *IEEE Sensors Journal*, vol. 14, no. 4, pp. 1124–1132, 2014.

[56] C. Hu, B. D. Youn, P. Wang, and J. Taek Yoon, "Ensemble of data-driven prognostic algorithms for robust prediction of remaining useful life," *Reliability Engineering & System Safety*, vol. 103, pp. 120–135, 2012.

[57] E. Ramasso and R. Gouriveau, "Remaining useful life estimation by classification of predictions based on a neuro-fuzzy system and theory of belief functions," *IEEE Transactions on Reliability*, vol. 63, no. 2, pp. 555–566, 2014.

[58] M. El-Koujok, R. Gouriveau, and N. Zerhouni, "Reducing arbitrary choices in model building for prognostics: an approach by applying parsimony principle on an evolving neuro-fuzzy system," *Microelectronics Reliability*, vol. 51, no. 2, pp. 310–320, 2011.

[59] K. Liu, N. Z. Gebraeel, and J. Shi, "A data-level fusion model for developing composite health indices for degradation modeling and prognostic analysis," *IEEE Transactions on Automation Science and Engineering*, vol. 10, no. 3, pp. 652–664, 2013.