Hindawi
International Journal of Aerospace Engineering
Volume 2023, Article ID 6610971, 18 pages
https://doi.org/10.1155/2023/6610971

Research Article

A Weight-Generating Approach of a Deep Neural Network for the Parameter Identification of Dynamic Systems

Weimeng Chu, Shunan Wu, Fangzhou Fu, Zhe Ye, and Zhigang Wu

School of Aeronautics and Astronautics, Shenzhen Campus of Sun Yat-sen University, Shenzhen 518107, China

Correspondence should be addressed to Shunan Wu; wushunan@mail.sysu.edu.cn

Received 2 February 2023; Revised 11 May 2023; Accepted 19 May 2023; Published 29 May 2023

Academic Editor: Jinchao Chen

The general learning process of deep learning is extremely time-consuming. Unlike the traditional learning process, a weight-generating approach to quickly generate the weight vectors of a deep neural network model is proposed, which can be used for parameter identification of a dynamic system. Based on the analysis of three trained deep neural network models, which are used to identify the parameters of three different dynamic systems, the statistical relationships between the weight vectors of each hidden layer and its inputs are revealed. Then, the statistical patterns of the weight vectors are imitated by exploiting the statistical patterns of the inputs and these relationships. Then, a weight-generating approach is designed to quickly generate the weight vectors of a deep neural network model. The effectiveness of the weight-generating approach is tested on the tasks of parameter identification for the three dynamic systems. The numerical results are provided to demonstrate the validity and high efficiency of the proposed weight-generating approach.

1. Introduction

The temporal evolution of dynamic systems is widely described using ordinary differential equations (ODEs), which are a set of governing differential equations. For many dynamic systems, such as aircraft and spacecraft attitude dynamics, ODEs can accurately represent the physical laws of the actual system and are very accurate [1, 2]. However, the system parameters in these ODEs are usually unknown and need to be determined according to the input and output data, i.e., the control and state data, of the dynamic systems.

Many studies have been proposed in the field of parameter identification [3, 4]. Most identification methods estimate the system parameters by minimizing the mean square error between the measured state data of a dynamic system and the numerical solution of its ODEs obtained using the estimated parameters. The least squares identification method is the most widely used parameter identification method in the aerospace field [5–10]. However, since the least squares identification method needs to use the state derivative, which is usually obtained by calculating the numerical differences on state data, it is very sensitive to the difference accuracy. To solve the abovementioned problem, some parameter identification methods that use a set of basis functions to represent state data have been proposed [11, 12]. These methods can use the analytic derivative of the basis function group instead of the numerical difference to calculate the state derivative. However, the identification accuracies of these methods are highly dependent on the selection of the type and number of basis functions. Another important parameter identification method is the Kalman filter [13–16]. The traditional Kalman filter can only be applied to linear systems, while the extended Kalman filter can be used for nonlinear systems. The extended Kalman filter needs to use linearization to approximate a nonlinear system, and the error introduced by linearization may lead the filter to diverge [17, 18]. Based on the above analysis, it can be found that most traditional identification methods need to solve the state derivative, which can easily introduce

errors and reduce the identification accuracy. Thus, some novel identification approaches should be developed for solving this problem.

With the development of deep learning (DL) in various fields [19–21], recent studies have attempted to expand DL applications to the field of parameter identification [22–25]. According to the Takens theorem, a large class of nonlinear dynamical systems can be effectively reconstructed using a sufficiently large system state and control sequence [26]. The system parameters are important features for describing the system dynamics. DL technology can learn the mapping relationship between the state and control sequence of a dynamic system and its corresponding parameters by automatically extracting multilayer abstract features implied in the system state and control sequence during training. Therefore, DL identification methods can use only the state and control sequence to identify the parameters of the dynamic system without the procedure of addressing state derivatives. In the general process of DL-based parameter identification, a deep neural network (DNN) model is first constructed with randomly generated initialization weight vectors. The weight vectors of the DNN model are then iteratively trained using training data. The commonly used methods to optimize weight vectors in the training process include the momentum method [27] and the adaptive moment (Adam) method [28]. There are also some optimization methods that can satisfy the Lyapunov stability to ensure the convergence of the training process [29–31]. But the acquisition of an accurate DNN model often takes a great deal of time.

Due to the high time consumption for training, some studies for a fast-learning approach of a neural network model have been explored. Random vector functional-link networks and extreme learning machines are widely used fast-learning approaches for neural networks with a single hidden layer, in which the weight vectors of the hidden layer are randomly generated without using any prior knowledge [32–34]. Nevertheless, DL is a set of hierarchical feature-learning methods that can automatically extract multilayer abstract features needed for target tasks from input data through training. The features at a higher layer of a DNN model are formed by transforming the features at the previous layer and then further amplifying important aspects of the input data and suppressing irrelative variation [35]. The transformation is implemented by inputting the linear combination of the features at the previous layer and the weight vectors of the DNN model at the current layer to nonlinear activation functions. The weight vectors of each layer of a DNN model can thus be regarded as a "filter" to the input of each layer. For example, an edge detection operator in image processing can be perceived as a filter that highlights the areas of an image where the grayscale changes rapidly. For parameter identification of a dynamic system, some statistical regularities and patterns in the state sequences that are important for parameter identification may exist due to the dynamic constraints. The weight vectors of a trained DNN model can be used as filters to extract these important statistical regularities and patterns from the state sequences, i.e., the inputs to the DNN model, and transform them into more abstract features layer-by-layer. Thus, the weight vectors of each layer of a DNN model should have some close relationships with their inputs.

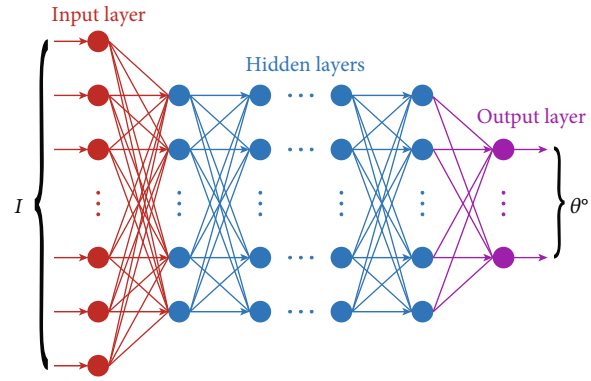

Figure 1: The structure of a fully connected DNN model used for parameter identification.

Based on the above considerations, this paper proposes a fast weight-generating approach to obtain the weight vectors of a DNN model used for parameter identification. By analyzing the trained DNN models for identifying the parameters of three dynamic systems, some statistical relationships between the weight vectors and inputs are discovered in each hidden layer. According to these relationships, a weight-generating approach is developed to generate the weight vectors by using the inputs of each hidden layer of the DNN model. The identification performance of the DNN model generated using the weight-generating approach is tested on the three dynamic systems. The results show the validity and high efficiency of this approach. More importantly, our work provides a way of understanding the operating mechanism for the DNN model, which can provide some ideas for further studies.

The rest of the paper is organized as follows. Section 2 introduces the general process of DL-based parameter identification. Detailed descriptions of the relationship analysis between the inputs and weight vectors of the trained DNN models and the design of the weight-generating approach are presented in Section 3. The effectiveness of the weight-generating approach is verified on the three dynamic systems in Section 4. Section 5 summarizes this work and discusses future work.

2. DL-Based Parameter Identification

2.1. Problem Statement. The ODEs of a dynamic system can be generally represented as

$$\begin{cases} \dfrac{d\mathbf{x}(t)}{dt} = f(\mathbf{x}(t), \mathbf{u}(t)|\boldsymbol{\theta}), & t \in [0, H], \\ \mathbf{x}(0) = \mathbf{x}_0, \end{cases} \tag{1}$$

where $\mathbf{x}(t) \in R^n$ is the state vector of the dynamic system at time t, $\mathbf{u}(t) \in R^r$ is the control vector, $f(\cdot)$ represents a nonlinear function for describing the evolution of the

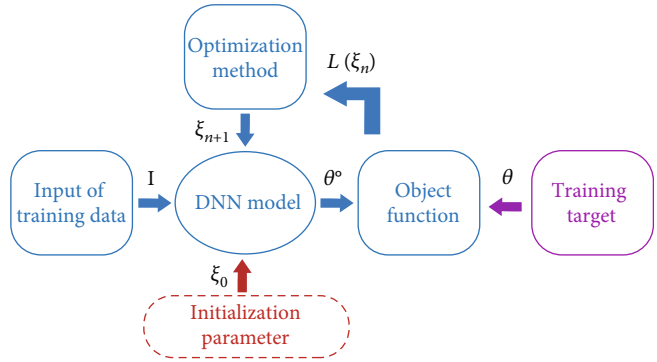

Figure 2: Schematic diagram of the training process.

system, and $\boldsymbol{\theta}$ denotes the unknown constant parameters in $f(\cdot)$, which need to be identified.

To use DL to identify the parameters of the dynamic system, it is necessary to determine the input and output of the DNN model. The Takens theorem states that for a large class of nonlinear dynamic systems with an N -dimensional state space, it can effectively reconstruct the system state to use $2N+1$ historical outputs of the system [26]. According to the Takens theorem, for finite-dimensional dynamic systems, when the historical state-control sequence $[\mathbf{x}(0:H);\mathbf{u}(0:H)]$ of the system is long enough, the reconstruction of the system state can be achieved, and the system parameters can also be effectively identified. Thus, the input of the DNN model is set to the state-control sequence $[\mathbf{x}(0:H);\mathbf{u}(0:H)]$, and the output is the parameter $\boldsymbol{\theta}$ to be identified.

The objective of constructing the DNN model is to enable the DNN model to learn the complex mapping relationship between state control sequences and system parameters in a specified dynamic system parameter space during the training process and then achieve accurate identification of any system parameters in the parameter space.

2.2. Data Generation. To make the DNN model to be suitable for the nonlinear system accurately, a large amount of data needs to be used for training. For this reason, a rough range of $\boldsymbol{\theta}$ can be empirically determined in practice and is used as a sampling dynamic system parameter space to generate data for training a DNN model. If the sampling space Γ of $\boldsymbol{\theta}$ is equal to or greater than the space of the actual parameters and the number of sample $\boldsymbol{\theta}$ is sufficient, a trained DNN model can effectively perform the identification task [22–24]. A subset Γ_s of $\boldsymbol{\theta}$ is then randomly selected from the sampling space Γ. According to Equation (1), a state sequence $\mathbf{x}_i(0:H)$ can be solved by a group of given $\boldsymbol{\theta}_i$ and control sequence $\mathbf{u}_i(0:H)$. $\mathbf{x}_i(0:H)$, $\mathbf{u}_i(0:H)$ and $\boldsymbol{\theta}_i$ constitute training data. $\mathbf{I}_i=[\mathbf{x}_i(0:H);\mathbf{u}_i(0:H)]^T$ is used as the input part of the training data, and $\boldsymbol{\theta}_i$ is used as its training target. If the control trajectory $\mathbf{u}_i(0:H)$ is predetermined and unchangeable for each sample, the input part can be simplified to $\mathbf{I}_i=[\mathbf{x}_i(0:H)]^T$. A training set is obtained by addressing all samples of Γ_s in this way.

2.3. DNN Model. The structure of the DNN model used for parameter identification is shown in Figure 1. It is a fully connected network, in which any neuron on any layer is connected to all neurons in the previous layer. The DNN model consists of one input layer, some hidden layers, and one output layer. The DNN identification model can be written as

$$\boldsymbol{\theta}^o = f_{\mathrm{dnn}}(\mathbf{I}|\boldsymbol{\xi}), \tag{2}$$

where $\boldsymbol{\theta}^o$ is the output of the DNN model, $\mathbf{I}$ is its input, and $\boldsymbol{\xi}$ represents the weight $\mathbf{w}$ and bias $\mathbf{b}$ within the DNN model.

Each neuron of all hidden layers is calculated as

$$\begin{cases} \alpha_k^l = \left(\boldsymbol{\beta}^l\right)^T \mathbf{w}_k^l + b_k^l, \\ \beta_k^{l+1} = \phi\left(\alpha_k^l\right), \end{cases} \tag{3}$$

where $\boldsymbol{\beta}^l$ is the input vector of the lth layer, which contains all outputs of the $l-1$th layer, $\mathbf{w}_k^l$, b_k^l are the weight vector and bias of the kth neuron of the lth layer, and $\phi(\cdot)$ is a nonlinear activation function that introduces a nonlinearity to each neuron.

The output layer is a linear computation

$$\boldsymbol{\theta}^o = \left(\boldsymbol{\beta}^o\right)^T \mathbf{W}^o + \mathbf{b}^o, \tag{4}$$

where $\boldsymbol{\beta}^o$ is the input vector of the output layer, which contains the full output of the last hidden layer, and $\mathbf{W}^o$, $\mathbf{b}^o$ are the weight matrix and bias vector of the output layer, respectively.

2.4. Training Process. A DNN model with a determined structure needs to be trained to learn the mapping relationship between the state and control sequential state $\mathbf{I}$ of a system and their corresponding parameters $\boldsymbol{\theta}$. A schematic diagram of the training process is shown in Figure 2.

The first step of the training process is to assign an initialization $\boldsymbol{\xi}_0$ to the DNN model. The input $\mathbf{I}$ of the training data is then input into the DNN model to obtain the output

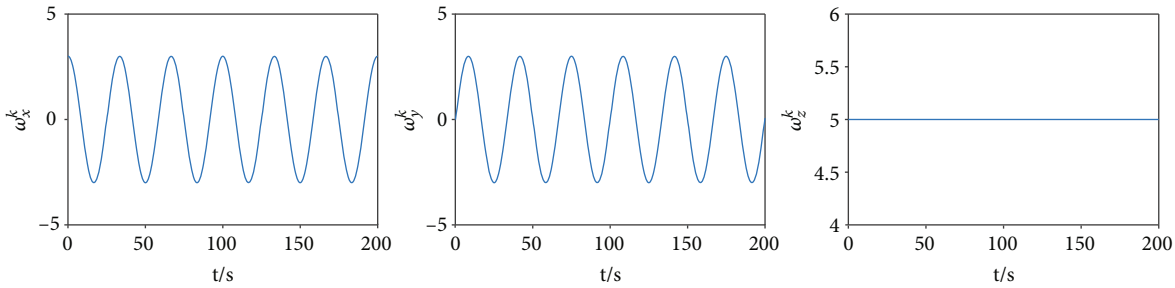

Figure 3: A group of angular velocity sequences selected randomly from the training set.

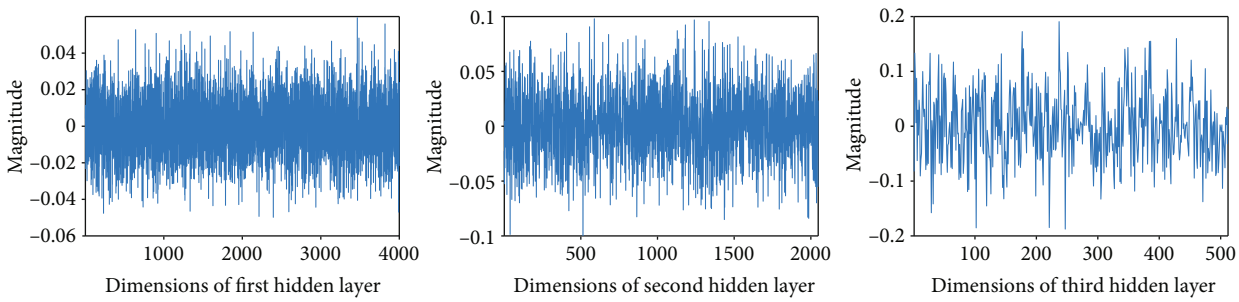

Figure 4: Visualization diagram of the weight vectors of the three neurons in model A.

$\boldsymbol{\theta}^o$. An objective function is given to measure the difference between the $\boldsymbol{\theta}^o$ and target $\boldsymbol{\theta}$. For parameter identification, the objective function is designed as

$$L(\boldsymbol{\xi}) = \frac{1}{D}\sum_{i=1}^{D}(\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^o)^2 = \frac{1}{D}\sum_{i=1}^{D}(\boldsymbol{\theta}_i - f_{\text{dnn}}(\mathbf{I}_i|\boldsymbol{\xi}))^2, \quad (5)$$

where D is the number of samples in the training set Γ_s. An optimization method, which is commonly selected from the gradient descent algorithm and its variants, is used to iteratively update $\boldsymbol{\xi}$ to minimize $L(\boldsymbol{\xi})$. Each optimization iteration contains two steps, the first step is to compute the gradient $\nabla L(\boldsymbol{\xi}_n)$, and then, $\boldsymbol{\xi}_n$ is updated by

$$\boldsymbol{\xi}_{n+1} = \boldsymbol{\xi}_n - \eta\nabla L(\boldsymbol{\xi}_n), \quad (6)$$

where η is the learning rate of training. After some epochs, a trained DNN model is obtained to identify any parameter in the space Γ.

3. Weight-Generating Approach of the DNN Model for Parameter Identification

3.1. Relationships between the Inputs and Weight Vectors of Each Hidden Layer. In this section, some relationships between the inputs and weight vectors of each hidden layer are revealed by analyzing the trained DNN models for identifying the parameters of three different dynamic systems. The detailed discussion is given below.

3.1.1. Attitude Dynamics of a Rigid Spacecraft. The attitude dynamics of a rigid spacecraft are first used for analysis. The adopted dynamic model is based on an axisymmetric rigid spacecraft, and its body coordinate *oxyz* is consistent with the direction of the principal axes. Suppose the moments of inertia J_x, J_y satisfy the relation $J_x = J_y = J_c$, i.e., the rigid spacecraft is axisymmetric about the spin axis *oz*, and $J_z > J_c$. In the absence of an external torque, the ODEs of the attitude dynamics are written as

$$\begin{cases} \dot{\omega}_x + \left(\dfrac{J_z}{J_c} - 1\right)\omega_z\omega_y = 0, \\ \dot{\omega}_z = 0, \\ \dot{\omega}_y - \left(\dfrac{J_z}{J_c} - 1\right)\omega_z\omega_x = 0, \end{cases} \quad (7)$$

where $\omega_x, \omega_y, \omega_z$ and $\dot{\omega}_x, \dot{\omega}_y, \dot{\omega}_z$ are the angular rate and the angular acceleration, respectively, of the body coordinate frame *oxyz* in an inertial reference frame.

A training set is generated by setting the initial value and range. The initial angular velocity $\boldsymbol{\omega}(0)$ is set to $\omega_x(0) = 3, \omega_y(0) = 0, \omega_z(0) = 5$, and the sampling range of J_x, J_y, J_z is [100, 120]. The simulation time H is 200 s, and the sampling frequency is 10 Hz. After solving Equation (7) based on the initial values and ranges, 2000 groups of data, which are generated by different J_x, J_y, J_z values, are used as the training set. To be convenient for training, each group of J_x^k, J_y^k, J_z^k is divided by 100 for the training target and has a new range of [1, 1.2]. Each data point consists of a three-dimensional angular velocity sequence $\mathbf{I}_\omega = [\boldsymbol{\omega}_x^k(0:H), \boldsymbol{\omega}_y^k(0:H), \boldsymbol{\omega}_z^k(0:H)]^T$ and the parameters J_x^k, J_y^k, J_z^k. Figure 3

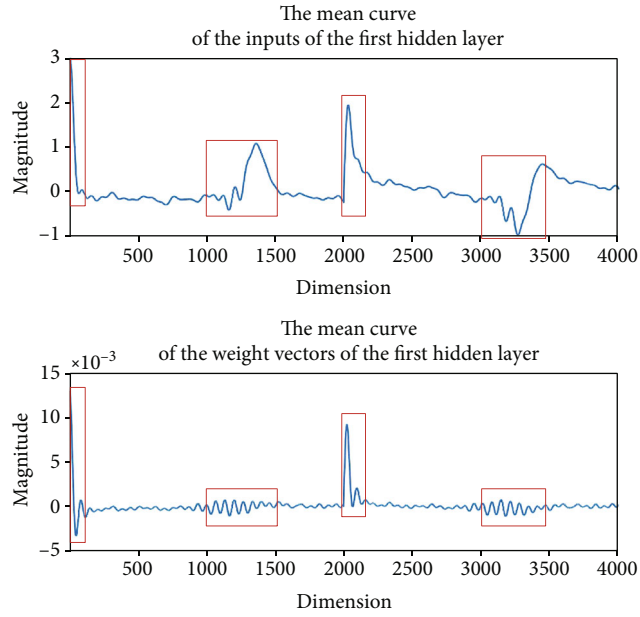

Figure 5: The mean curves of the first hidden layer of model A-L2.

shows a group of angular velocity sequences selected randomly from the training set. $\boldsymbol{\omega}_x^k(0:H), \boldsymbol{\omega}_y^k(0:H)$ are periodic curves, and $\boldsymbol{\omega}_z^k(0:H)$ is constant. Due to $\dot{\omega}_z = 0$, all points of $\boldsymbol{\omega}_z^k(0:H)$ are always equal to the initial value $\omega_z(0) = 5$ for all data. The input part of the training data is simplified to $\mathbf{I}_\omega = [\boldsymbol{\omega}_x^k(0:H), \boldsymbol{\omega}_y^k(0:H)]^T$. In addition, a testing set with 100 groups of data that are different from the data of the training set is generated in this way to test model accuracy.

The structure of the DNN identification model is determined to have three hidden layers with 2048, 512, and 128 hidden neurons. A rectified linear unit (ReLU) is proposed as the activation function, and its expression could be written as $\phi(x) = \max(0, x)$ [36]. The weight $\mathbf{w}$ is initialized using the Microsoft Research Asia (MSRA) method, and the bias $\mathbf{b}$ is set to 0 [37]. The used optimization method is the adaptive moment (Adam) estimation method, which can adaptively adjust the learning rate by estimating the first-order and second-order moments of the gradient in the objective function to accelerate the training efficiency of the network [38]. For the selections of hyperparameters, the learning rate is 1×10^{-4}, and the size of a minibatch is 50. After 2000 optimization iterations, a trained DNN identification model is obtained and is called model A. The mean square error (MSE) of model A for identifying the test data is 5.20×10^{-5}.

Figure 4 shows a visualization diagram of the weight vectors of the three neurons in model A. These neurons are randomly selected from the three different hidden layers. From Figure 4, the weight vector of each neuron is analogous to a random distribution; hence, it is difficult to determine the relationship between the distribution of the weight vectors and the distribution of the inputs to the DNN model.

To make the weight vectors more discriminative, an L2-regularization term is added to the object function

$$L(\boldsymbol{\xi}) = \frac{1}{D}\sum_{i=1}^{D}(\boldsymbol{\theta}_i - f_{\text{dnn}}(\mathbf{I}_i|\boldsymbol{\xi}))^2 + \lambda\sum_{j=1}^{B} w_j^2, \tag{8}$$

where λ is the regularization coefficient and B is the number of model weight vectors. The L2-regularization term can shrink model weight vectors toward 0, which makes the weight matrix sparse in each layer and reduces the complexity of the DNN model. The regularization coefficient is set to 0.001 during training. For the learning rate, it is difficult to find an optimal value. If the learning rate is too large, the training result may stay in a poor local minimum, which leads to a poor convergence accuracy, while it will cause slow convergence and long training time for a small learning rate [39]. Thus, a decaying learning rate is selected to train the DNN model, i.e., the learning rate is larger in the initial stage and gradually decreases as the training progresses, which could improve the convergence speed. The learning rate is set to 1×10^{-2} and is multiplied by 0.2 after every 15,000 iterations. A trained model called model A-L2 is obtained after 60,000 optimization iterations. The weight matrix of each layer of model A-L2 has been effectively compressed. Only 30 of the 2048 weight vectors of the first hidden layer, 53 of the 512 weight vectors of the second layer, and 16 of the 128 weight vectors of the third layer are nonzero vectors. The MSE of model A-L2 for identifying the test data is 1.38×10^{-5} and is superior to model A, which indicates that the weight vectors of model A-L2 extract the important information used for identification that is implicit in the weight vectors of model A.

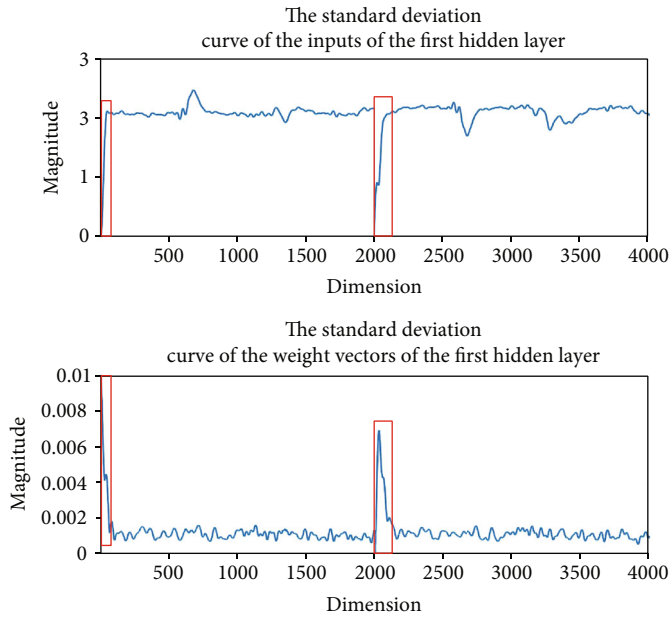

Figure 6: The standard deviation curves of the first hidden layer of model A-L2.

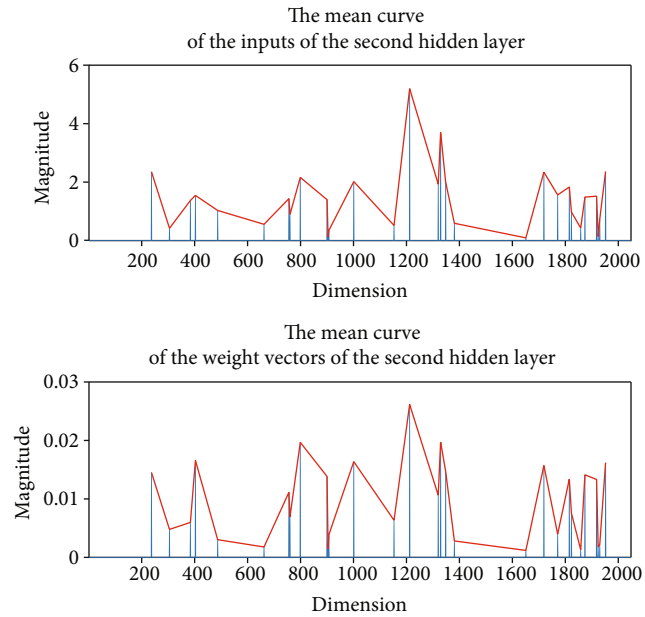

Figure 7: The mean curves of the second hidden layer of model A-L2.

To search the relationship between the inputs and weight vectors of the DNN model, the statistical characteristics of model A-L2 are computed for analysis. The mean curves and standard deviation curves of the inputs and weight vectors of the first hidden layer are shown in Figures 5 and 6, respectively. The two statistical curves of the inputs are obtained by calculating the mean and standard deviation of each angular velocity sequence point in the input parts of all training data. According to Equation (3), the input to each layer is combined with a weight vector to compute the inner product. Thus, based on the inner product calculation, the sequence order of the statistical curves of the weight vectors is consistent with that of the corresponding statistical curves of the inputs. As Figures 5 and 6 show, the parts in the red boxes in the statistical curves of the inputs have more rapid change rates than other parts, while the statistical curves of the weight vectors highlight the corresponding parts in its curves and suppress most other parts. This illustrates that the statistical patterns of the weight vectors of the first hidden layer are related to the change rates of the statistical patterns of the inputs.

The second hidden layer receives the outputs of the first hidden layer and further extracts features used for parameter identification. Figures 7 and 8 show the mean

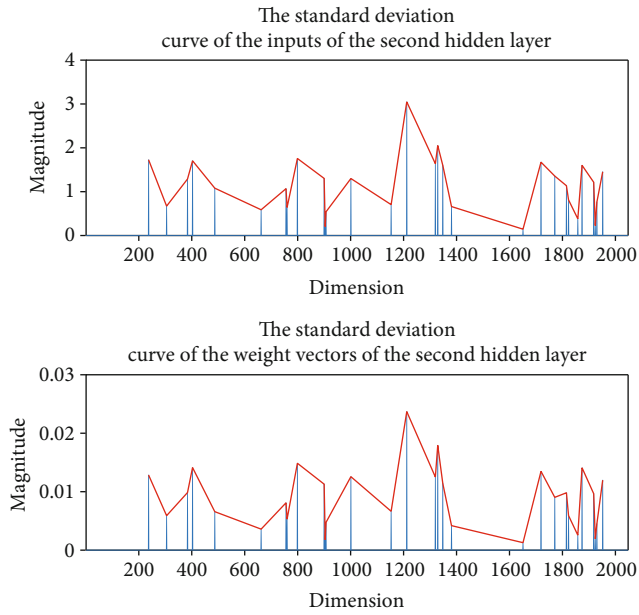

Figure 8: The standard deviation curves of the second hidden layer of model A-L2.

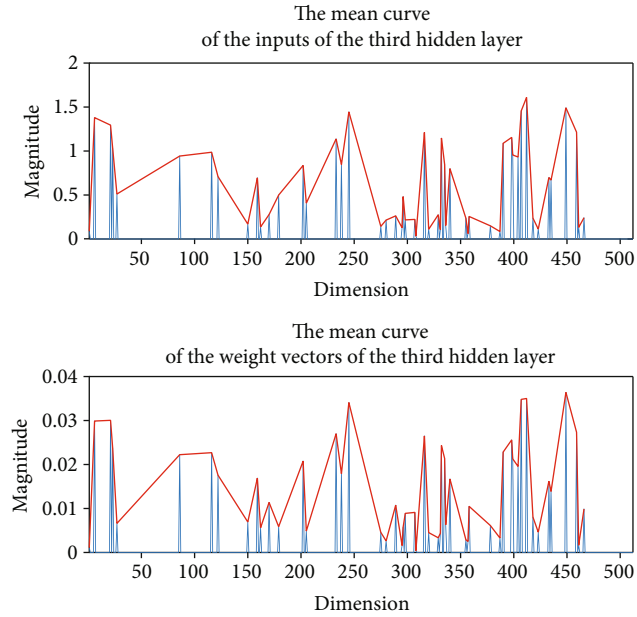

Figure 9: The mean curves of the third hidden layer of model A-L2.

curves and standard deviation curves of the inputs and weight vectors, respectively, of the second hidden layer. As shown in Figures 7 and 8, all the statistical curves are composed of discretized points. By connecting the discretized points using red line segments, the variation trends of different statistical curves are compared by judging the monotonicity of these line segments. The variation trends for the two statistical curves of the inputs are almost the same as those for the two curves of the weight vectors. It should be noted that the mean curve and standard deviation curve of the weight vectors have nearly the same overall magnitude. This phenomenon also appears in the third hidden layer. The mean curves and standard deviation curves of the inputs and weight vectors of the third hidden layer are presented in Figures 9 and 10, respectively. As Figures 9 and 10 show, 88.5% and 76.9% of the variation trends of the mean curve and standard deviation curve, respectively, of the inputs are the same as those of the two curves of the weight vectors. The mean curve and standard deviation curve of the weight vectors of the third hidden layer also have nearly the same overall magnitude. This shows that the inputs and weight vectors

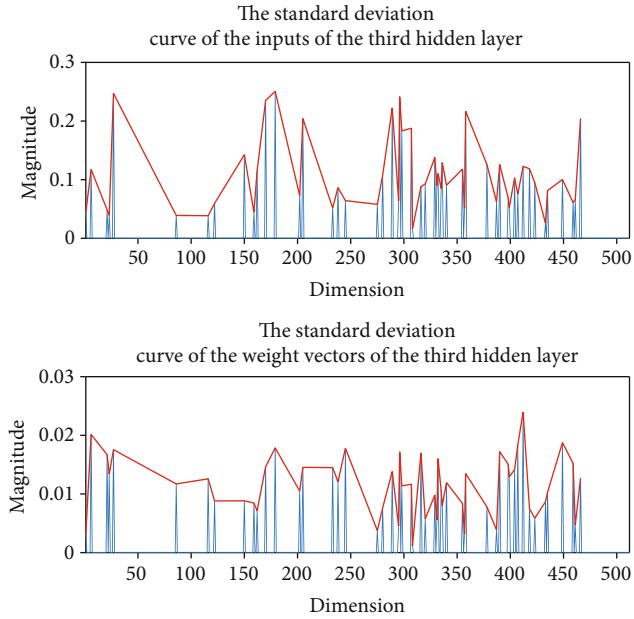

Figure 10: The standard deviation curves of the third hidden layer of model A-L2.

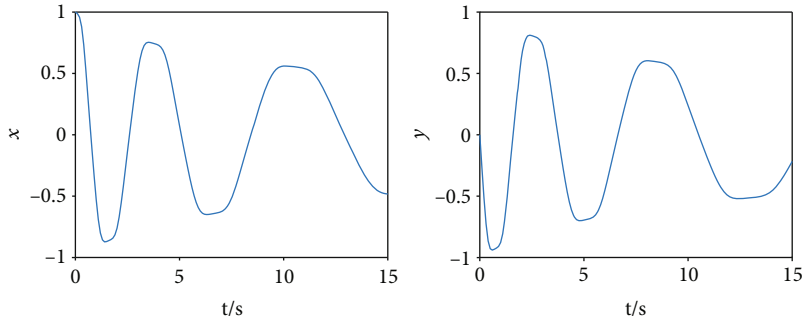

Figure 11: A state sequence of damped harmonic oscillator selected randomly from the training set.

of each of the last two hidden layers have similar statistical patterns, and the two statistical curves of the weight vectors of each layer have nearly the same overall magnitude.

3.1.2. Two-Dimensional Damped Harmonic Oscillator. A two-dimensional damped harmonic oscillator is selected to further verify the generalization of the relationships above. The ODEs of the system dynamics are written as follows [40]:

$$\begin{cases} \dot{x} = -ax^3 + by^3, \\ \dot{y} = -bx^3 - ay^3. \end{cases} \tag{9}$$

To generate a training set, the initial state is set to $x(0) = 1, y(0) = 0$, and the sampling ranges of a and b are [0.1, 0.5] and [1, 5], respectively. The simulation time H is 15 s, and the sampling frequency is 10 Hz. According to Equation (9), 2000 groups of data, which are generated using different a, b, are used as the training set. Each pair of a^k, b^k is normalized in a new range of [0.2, 0.8] and is used as the training target. Each data point includes a two-dimensional state sequence $\mathbf{I}_d = [\mathbf{x}^k(0:H), \mathbf{y}^k(0:H)]^T$ and the parameters a^k, b^k. Figure 11 shows a state sequence selected randomly from the training set. Likewise, an extra testing set with 100 groups of data is generated to test model accuracy.

The structure of the DNN identification model is determined to have two hidden layers with 64 and 128 hidden neurons. The selections for the activation function, initialization method, and optimization method are consistent with those used for training model A. The learning rate is 1×10^{-3}, and the size of a minibatch is 50. After 5000 optimization iterations, a trained DNN identification model called model B is obtained. The MSE of model B for identifying the test data is 2.28×10^{-4}.

Another DNN model trained on the objective function with an L2-regularization term is also obtained to analyze the relationships between the inputs and weight vectors of

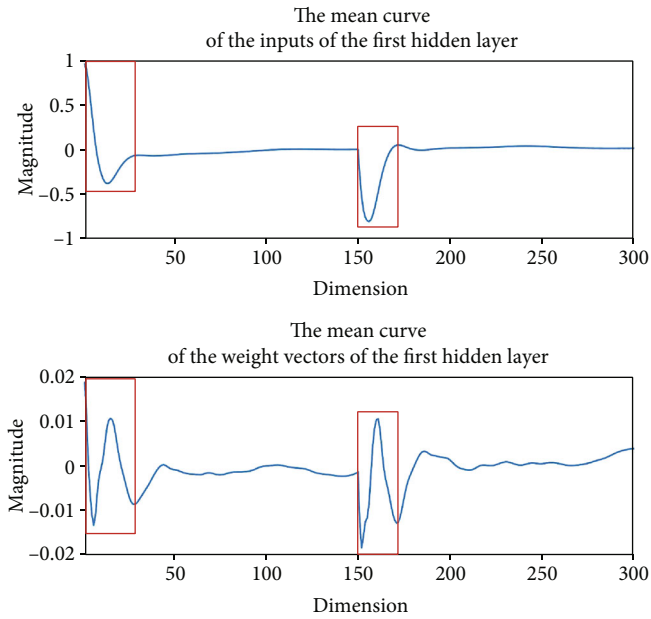

Figure 12: The mean curves of the first hidden layer of model B-L2.

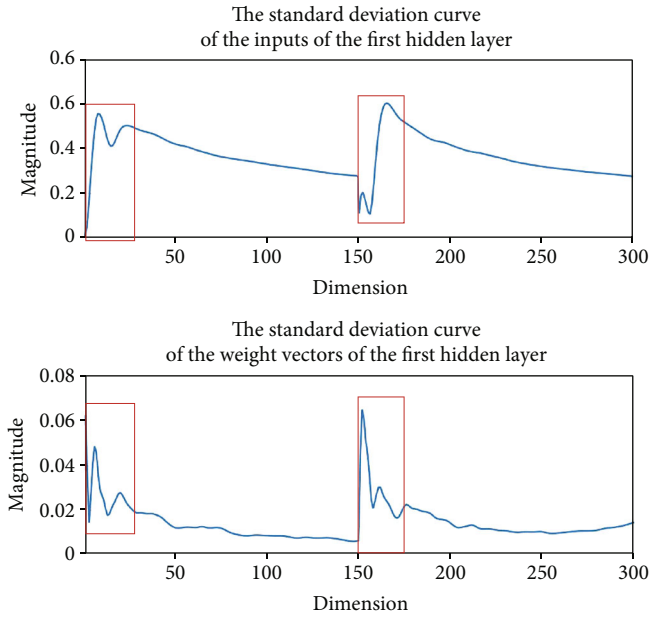

Figure 13: The standard deviation curves of the first hidden layer of model B-L2.

each layer. The regularization coefficient is set to 0.001 during training, and the learning rate is set to 1×10^{-2} and is multiplied by 0.2 after every 20,000 iterations. After 80,000 optimization iterations, a trained model called model B-L2 is obtained. In this case, 21 of the 64 weight vectors of the first hidden layer and 21 of the 128 weight vectors of the second layer are nonzero vectors. The MSE of model B-L2 for identifying the test data is 2.58×10^{-4} and is quite close to that of model B.

The statistical characteristics of model B-L2 are then computed for analysis. Figures 12 and 13 show the mean curves and standard deviation curves of the inputs and weight vectors, respectively, of the first hidden layer. It can be seen that the relationship between the statistical patterns of the weight vectors and the change rates of the statistical patterns of the inputs are also similar to the previous analysis. The statistical curves of the weight vectors of the first hidden layer highlight the parts of the statistical curves of the inputs that have more rapid change rates and suppress most other parts.

Figures 14 and 15 show the mean curves and standard deviation curves of the inputs and weight vectors, respectively, of the second hidden layer. The variation trends for the two statistical curves of the inputs are also similar to

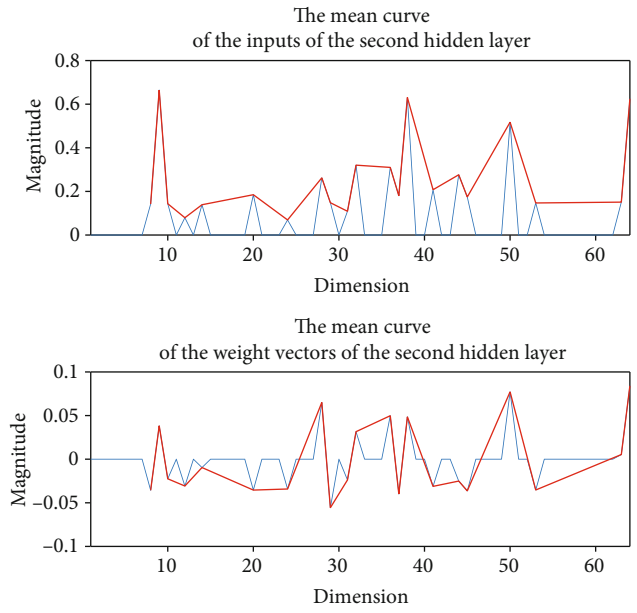

Figure 14: The mean curves of the second hidden layer of model B-L2.

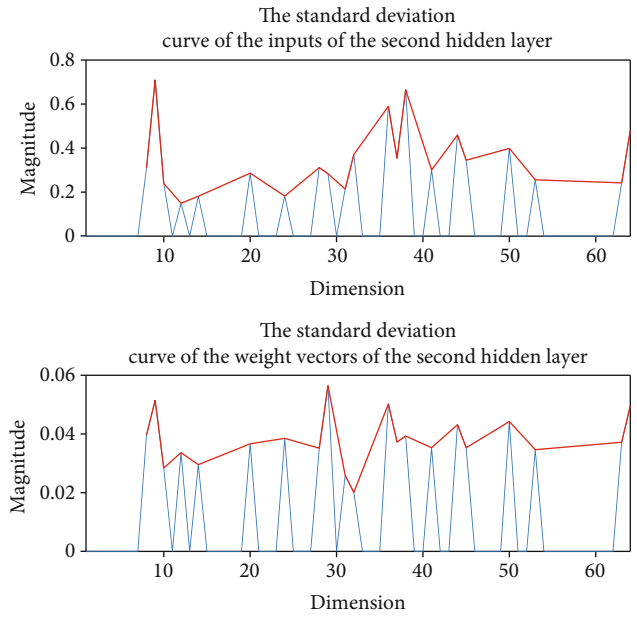

Figure 15: The standard deviation curves of the second hidden layer of model B-L2.

those for the two curves of the weight vectors. By judging the monotonicity of the red line segments, 80.0% and 65.0% of the variation trends of the mean curve and standard deviation curve, respectively, of the inputs are the same as those of the two curves of the weight vectors. The two statistical curves of the weight vectors also have approximately the same overall magnitude.

3.1.3. Damped Pendulum. The third dynamic system used for analysis is a damped pendulum. The ODE of the system dynamics is written as

$$\ddot{x} = -\alpha x - \beta \dot{x}. \tag{10}$$

By substituting $\dot{x}$ for y, Equation (10) is rewritten as

$$\begin{cases} \dot{x} = y, \\ \dot{y} = -\alpha x - \beta y. \end{cases} \tag{11}$$

The initial state is set to $x(0) = 1, y(0) = 0$. The sampling ranges of α and β are [5, 10] and [0.5, 1], respectively. The simulation time H is 25 s, and the sampling frequency is 10 Hz. According to Equation (11), 2000 groups of data are generated as a training set, and 100 groups are generated as a testing set. α^k is then normalized in a new range of [0.5, 1]. Each data point includes a two-dimensional state

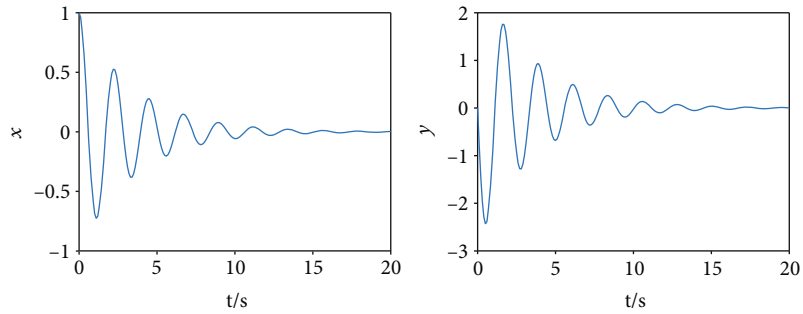

Figure 16: A state sequence of damped pendulum selected randomly from the training set.

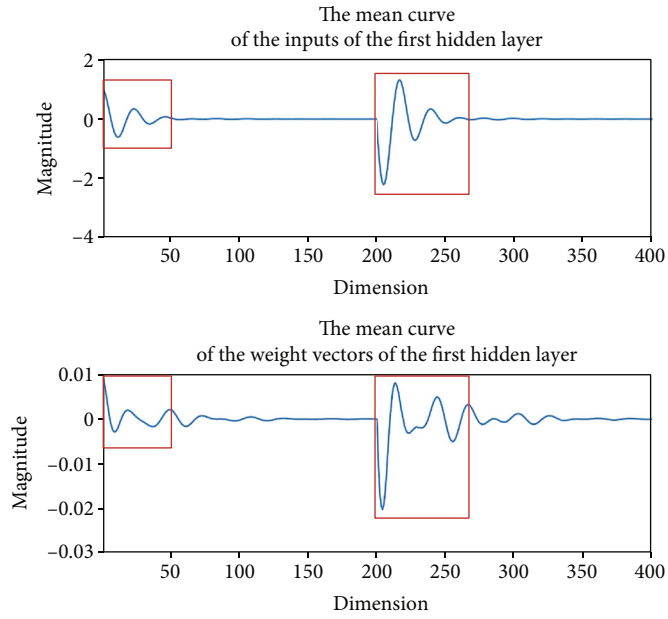

Figure 17: The mean curves of the first hidden layer of model C-L2.

sequence $\mathbf{I}_p = [\mathbf{x}^k(0:H), \mathbf{y}^k(0:H)]^T$ and the parameters α^k, β^k. Figure 16 shows a state sequence selected randomly from the training set.

The DNN model for this identification task has two hidden layers with 128 and 16 hidden neurons. The selections for the activation function, initialization method, optimization method, and hyperparameters are consistent with those used for training model B. After 5000 optimization iterations, a trained model, model C, is obtained. The MSE of model C for identifying the test data is 3.41×10^{-5}.

A DNN model trained on the objective function with an L2-regularization term is then obtained. The regularization coefficient is set to 0.001 during training. The learning rate is set to 1×10^{-2} and is multiplied by 0.2 after every 10,000 iterations. The trained model, model C-L2, is obtained after 40,000 optimization iterations. In this case, 53 of the 128 weight vectors of the first hidden layer and 8 of the 16 weight vectors of the second layer are nonzero vectors. The MSE of model C-L2 for identifying the test data is 8.96×10^{-5}. Figures 17 and 18 show the mean curves and standard deviation curves of the inputs and weight vectors, respectively, of the first hidden layer. Figures 19 and 20 show the mean curves and standard deviation curves of the inputs and weight vectors, respectively, of the second hidden layer. The relationships between the inputs and weight vectors of each layer are consistent with the previous analysis. The statistical curves of the weight vectors of the first hidden layer highlight the parts of the statistical curves of the inputs that have more rapid change rates and suppress most other parts. The statistical curves of the weight vectors of the second hidden layer have approximately consistent trends with the statistical curves of the inputs. As Figures 19 and 20 show, 76.9% and 75.0% of the variation trends for the mean curve and standard deviation curve, respectively, of the inputs are the same as those for the two curves of the weight vectors. For the mean curve and standard deviation curve of the weight vectors, their overall magnitudes are also quite close.

3.2. Weight-Generating Approach. According to the analysis above, there exist close relationships between the inputs and weight vectors of each hidden layer of a DNN model. There should be a way to directly transform the inputs of each hidden layer into the weight vectors of the corresponding layer by imitating these relationships to construct a DNN model, which can achieve high accuracy of parameter identification

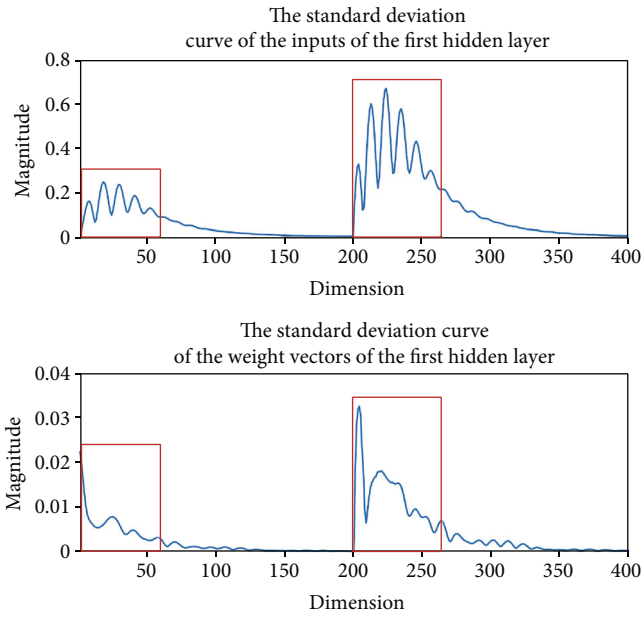

Figure 18: The standard deviation curves of the first hidden layer of model C-L2.

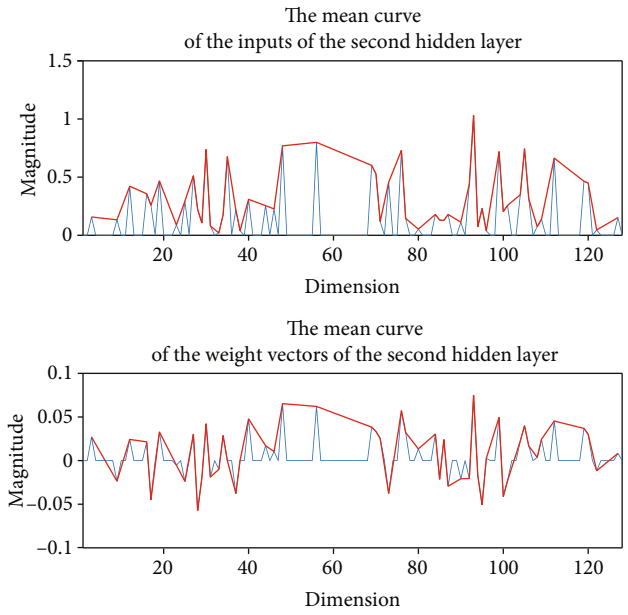

Figure 19: The mean curves of the second hidden layer of model C-L2.

without traditional training processes. Based on this idea, a weight-generating approach of a DNN model for parameter identification is proposed.

The weight vectors of the first hidden layer are designed to combine the statistical patterns of the inputs with the change rates of their statistical patterns. The design goal is to highlight the parts with rapid change rates in the two statistical curves of the inputs to the first hidden layer and suppress the parts with slower change rates. The change rates of the mean curve and standard deviation curve of the inputs are obtained by calculating the absolute differences of the two statistical curves. The mean curve and standard deviation curve of the inputs to the first hidden layer are represented as

$$\begin{cases} \text{mean}^1 = \left[\text{mean}_1^1(1:L), \text{mean}_2^1(1:L), \cdots, \text{mean}_N^1(1:L)\right]^T, \\ \text{std}^1 = \left[\text{std}_1^1(1:L), \text{std}_2^1(1:L), \cdots, \text{std}_N^1(1:L)\right]^T, \end{cases} \tag{12}$$

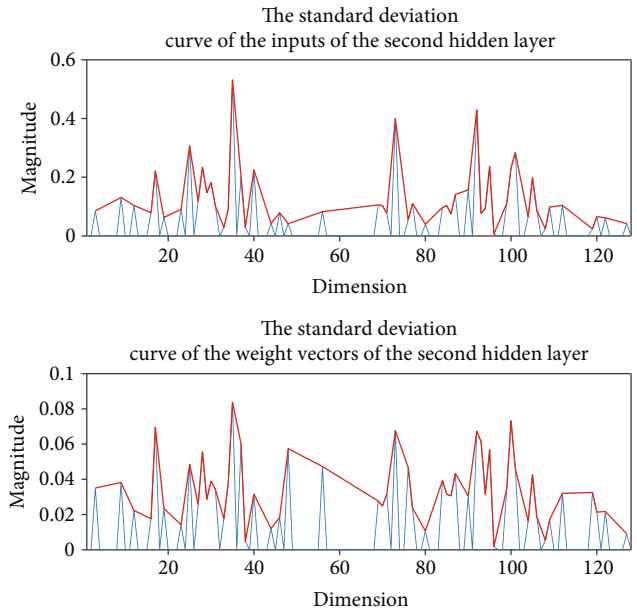

FIGURE 20: The standard deviation curves of the second hidden layer of model C-L2.

where N is the dimensions of the input to the first hidden layer, for example, the angular rate sequence $[\boldsymbol{\omega}_x^k(0:H), \boldsymbol{\omega}_y^k(0:H)]$ contains two dimensions, and L is the number of sequence points of each dimension. The absolute difference of each dimensional sequence of the statistical curve is calculated.

$$\begin{cases} \Delta\text{mean}_n^1(i) = \left|\text{mean}_n^1(i) - \text{mean}_n^1(i-1)\right|, \\ \Delta\text{std}_n^1(i) = \left|\text{std}_n^1(i) - \text{std}_n^1(i-1)\right|, \\ \quad i = 2, 3, \cdots, L, \\ \quad n = 1, 2, \cdots, N, \end{cases} \tag{13}$$

where $\text{mean}_n^1(i)$ and $\text{std}_n^1(i)$ are the ith point of the nth dimensional sequence in the mean curve and standard deviation curve, respectively, of the first hidden layer. All changing rates are then normalized as

$$\begin{cases} \Delta N\text{mean}_n^1(i) = \dfrac{\Delta\text{mean}_n^1(i)}{\sum_{i=2}^{L}\Delta\text{mean}_n^1(i)}, \\ \Delta N\text{std}_n^1(i) = \dfrac{\Delta\text{std}_x^1(i)}{\sum_{i}^{L}\Delta\text{std}_n^1(i)}, \\ \quad i = 2, 3, \cdots, L, \\ \quad n = 1, 2, \cdots N. \end{cases} \tag{14}$$

Two normalized changing rate sequences are obtained

$$\begin{cases} \Delta N\text{mean}^1 = \left[\left[\Delta N\text{mean}_1^1(2), \Delta N\text{mean}_1^1(2:L)\right], \left[\Delta N\text{mean}_2^1(2), \Delta N\text{mean}_2^1(2:L)\right], \cdots \left[\Delta N\text{mean}_N^1(2), \Delta N\text{mean}_N^1(2:L)\right]\right]^T, \\ \Delta N\text{std}^1 = \left[\left[\Delta N\text{std}_1^1(2), \Delta N\text{std}_1^1(2:L)\right], \left[\Delta N\text{std}_2^1(2), \Delta N\text{std}_2^1(2:L)\right], \cdots \left[\Delta N\text{std}_N^1(2), \Delta N\text{std}_N^1(2:L)\right]\right]^T. \end{cases} \tag{15}$$

To further highlight the parts with rapid change rates, the statistical patterns of the weight vectors of the first hidden layer are imitated by calculating the dot product between the statistical patterns of the inputs and the normalized changing rate sequences

$$\begin{cases} \text{PImean}^1 = \left(\Delta N\text{mean}^1\right)^T \text{mean}^1, \\ \text{PIstd}^1 = \left(\Delta N\text{std}^1\right)^T \text{std}^1, \end{cases} \tag{16}$$

where PImean^1 and PIstd^1 are the imitated mean vector and standard deviation vector, respectively, of the weight vectors of the first hidden layer.

The weight vectors of the first hidden layer are thus generated based on PImean^1 and PIstd^1. The sequence point $w_k^1(j)$ of each weight vector $\mathbf{w}_k^1$ is sampled from a Gaussian distribution $N(\text{PImean}^1(j)/P, \text{PIstd}^1(j))$, in which the constant P is used to adjust the scale of PImean^1 to ensure a close magnitude with PIstd^1. All weight vectors of the first

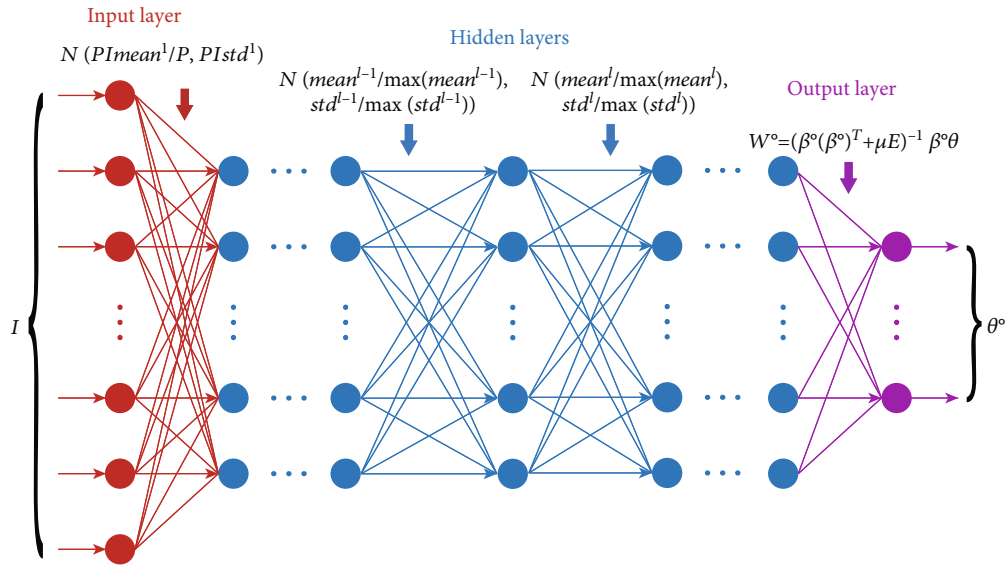

Figure 21: Schematic diagram of the weight-generating approach.

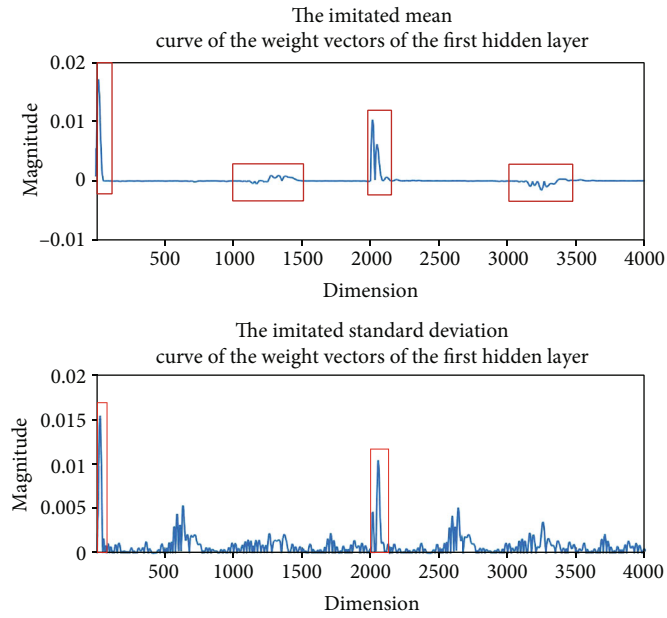

Figure 22: The imitated mean curve and standard deviation curve of the first hidden layer of model A-F.

hidden layer are generated in this way and are used to compute the inputs to the second hidden layer according to Equation (3).

For the other hidden layers, since their inputs and weight vectors have similar statistical patterns, each weight vector $\mathbf{w}_k^l$ of the lth hidden layer is generated by sampling from N (mean^l/max (mean^l),std^l/max (std^l)), where mean^l and std^l represent the mean vector and standard deviation vector, respectively, of the inputs to the th hidden layer. max (mean^l) and max (std^l) are the max values in mean^l and std^l, which are used to make the magnitudes of the mean vector and standard deviation vector of the weight vectors consistent.

Table 1: Comparison of the identification accuracies for the testing set of attitude dynamics using different DNN models.

	Model A	Model A-L2	Model A-R	Model A-F
MSE	5.20×10^{-5}	1.38×10^{-5}	1.26×10^{-3}	1.93×10^{-5}

The operational method of the output layer is linear; hence, its weight matrix $\mathbf{W}^o$ can be obtained using the ridge regression method [34].

$$\mathbf{W}^o = \left(\boldsymbol{\beta}^o(\boldsymbol{\beta}^o)^T + \mu\mathbf{E}\right)^{-1}\boldsymbol{\beta}^o\boldsymbol{\theta}, \tag{17}$$

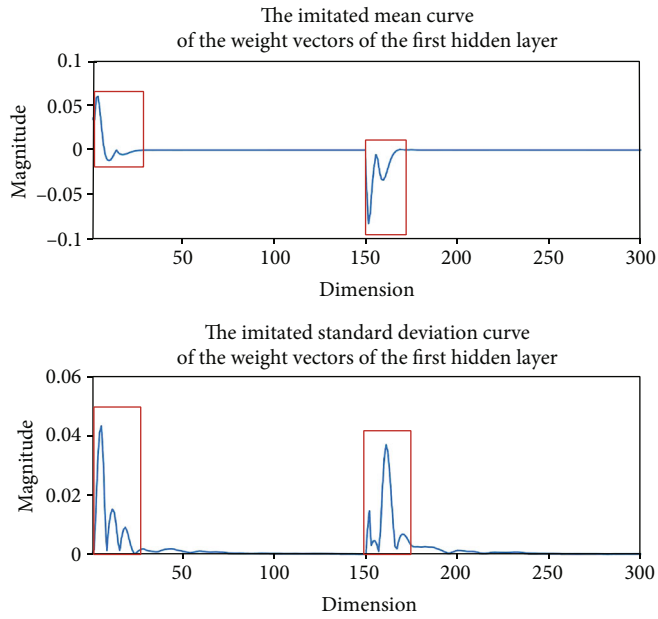

Figure 23: The imitated mean curve and standard deviation curve of the first hidden layer of model B-F.

where μ is the regularization coefficient which is used to mitigate the problem of multicollinearity and $\mathbf{E}$ is an identity matrix. A schematic diagram of the weight-generating approach of the DNN model for parameter identification is shown in Figure 21.

4. Simulation and Analysis

4.1. Attitude Dynamics of a Rigid Spacecraft. The task of identifying the parameters of the attitude dynamics is first applied to verify the performance of the DNN model using the weight-generating approach. The DNN model A-F is set to have the same structure as model A. The hyperparameters, used to construct model A-F, are set to $P = 5$ and $\mu = 1 \times 10^{-3}$. According to Equations (11)-(15), the imitated mean curve and standard deviation curve of the weight vectors of the first hidden layer are obtained and are shown in Figure 22. As shown in Figure 22, the two statistical curves have quite similar trends with the statistical curves of the weight vectors in Figures 5 and 6, which highlight the parts in the red boxes and suppress most other parts.

To verify the validity of the weight generating approach, the identification accuracy of model A-F is compared with model A and model A-L2, which are trained using the Adam method. And an extra DNN model is trained based on the extreme training machine, model A-R, whose weight vectors of the three hidden layers are all generated by sampling from a Gaussian distribution $N(0, 1)$ and the weight matrix of the output layer is also obtained according to Equation (17). In this work, all simulations are conducted on a laptop with an Intel Core i7-6820HK CPU and 32 GB memory.

Table 1 shows the comparison of the identification accuracies for the testing set using the different DNN models. As shown in Table 1, model A-F obtains a high identification accuracy. The identification accuracy of model A-F is superior to that of model A, which is trained on 2000 iterations, and is close to that of model A-L2, which is trained on 60,000 iterations. The identification accuracy of model A-R is the worst. The time to build model A-F is approximately 0.8 s, while the times for training model A and model A-L2 are approximately 19 min and 9 hours, respectively. This illustrates that the relationships between the inputs and the weight vectors of each hidden layer exist and can be exploited to rapidly build a DNN model with a high identification accuracy.

Table 2: Comparison of the identification accuracies for the testing set of damped harmonic oscillator using different DNN models.

	Model B	Model B-L2	Model B-R	Model B-F
MSE	2.28×10^{-4}	2.58×10^{-4}	2.05×10^{-3}	2.67×10^{-4}

4.2. Two-Dimensional Damped Harmonic Oscillator. The weight-generating approach is then applied to generate a DNN model used for identifying the parameters of a two-dimensional damped harmonic oscillator. The structure of the DNN model is the same as that of model B. The hyperparameters used for building model B-F are set to $P = 1$ and $\mu = 1 \times 10^{-6}$. Likewise, the imitated mean vector and standard deviation vector of the weight vectors of the first hidden layer are obtained according to Equations (11)-(15) and are shown in Figure 23. As Figure 23 shows, the trends of the two imitated statistical curves are also similar to the statistical curves of the weight vectors in Figures 12 and 13.

The identification accuracy of model B-F is also compared with three DNN models, model B and model B-L2 trained using the Adam method. And an extra DNN model is trained based on the extreme training machine, model B-R, whose weight vectors for the two hidden layers are all

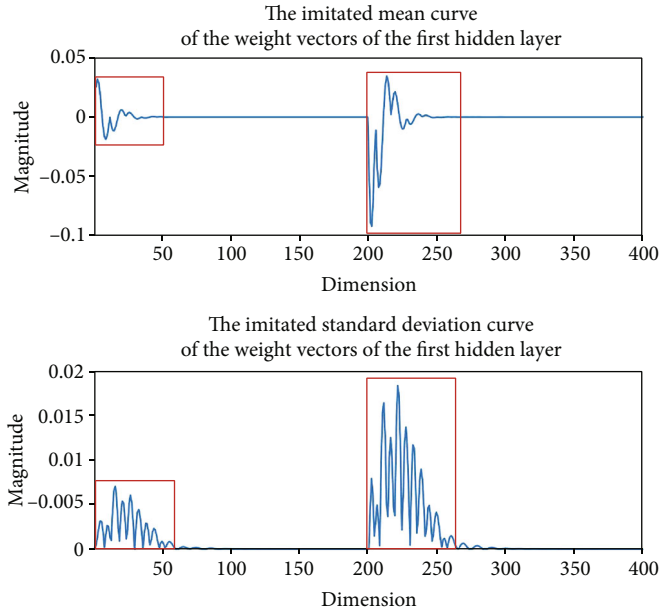

Figure 24: The imitated mean curve and standard deviation curve of the first hidden layer of model C-F.

generated by sampling from a Gaussian distribution $N(0, 1)$, and the weight matrix for the output layer is obtained according to Equation (17). Table 2 shows the comparison of the identification accuracies for the testing set using the different DNN models. As Table 2 shows, the identification accuracy of model B-F is still quite high. The identification accuracy of model B-R is the worst. The time to build model B-F is approximately 0.02 s, while the times for training model B and model B-L2 are approximately 4 min and 1 hour, respectively.

4.3. Damped Pendulum. A DNN model, model C-F, is also obtained using the weight generating approach. The structure of model C-F is the same as that of model C. The hyperparameters used for building model C-F are set to $P = 1$ and $\mu = 1 \times 10^{-5}$. The imitated mean curve and standard deviation curve of the weight vectors of the first hidden layer are shown in Figure 24. The two imitated statistical curves also highlight the parts in the red boxes and suppress the other parts.

The comparison of identification accuracies for the testing set among model C-F, model C, model C-L2, and an extra DNN model trained based on the extreme training machine, model C-R, whose weight vectors for the two hidden layers are all generated by sampling from a Gaussian distribution $N(0, 1)$ and weight matrix for the output layer is obtained according to Equation (17), is shown in Table 3. Model C and model C-L2 are trained using the Adam method. As Table 3 shows, the identification accuracy of model C-F is significantly better than that of the other DNN models. The time to build model C-F is approximately 0.017 s, while the times for training model C and model C-L2 are approximately 5 min and 40 min, respectively. This further proves the validity and high efficiency of the weight-generating approach of a DNN model for parameter identification. Furthermore, the analysis results illustrate that the change rates of the statistical patterns of the state sequences are important information for the parameter identification tasks of dynamic systems based on a DL method.

Table 3: Comparison of the identification accuracies for the testing set of damped pendulum using different DNN models.

	Model C	Model C-L2	Model C-R	Model C-F
MSE	3.41×10^{-5}	8.96×10^{-5}	1.91×10^{-3}	5.10×10^{-6}

5. Conclusion

In this paper, a weight-generating approach is designed to directly build a DNN model, which is used for identifying the parameters of dynamic systems. This paper analyzes the trained DNN models that are used to identify the parameters of three different dynamic systems and reveals some relationships between the inputs and weight vectors of the hidden layers in the tasks of parameter identification. The analysis results show that the statistical patterns of the weight vectors of the first hidden layer are related to the change rates of the statistical patterns of the inputs, while the weight vectors of each of the other hidden layers have a statistical pattern similar to their inputs. These relationships are utilized to design the weight generating approach. The performances of the DNN models generated using the weight-generating approach for identifying the parameters of the three dynamic systems are compared with different DNN models. The comparison results illustrate the validity and efficiency of the weight-generating approach for the task of parameter identification.

This work also provides insight for understanding the internal operating mechanisms of DNN models used for the parameter identifications of dynamic systems. Nevertheless, due to the complexity of the internal operating mechanisms of DNN models, there still exist other unknown and important information hidden in the weight vectors of DNN models. In future work, we will further analyze the internal operating mechanisms of DNN models to gain more insight and support theories that can be leveraged to improve the application capability of a DNN model in the field of parameter identification.

Data Availability

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

Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (91748203 and 11972102), the Shenzhen Science and Technology Program (Grant Nos. 202206193000001 and 20220816231330001), and the State Key Laboratory of Robotics and Systems (HIT) (SKLRS-2022-KF-08).

References

[1] Y. Barsinai, S. Hoyer, J. Hickey, and M. P. Brenner, "Learning data-driven discretizations for partial differential equations," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 116, no. 31, pp. 15344–15349, 2019.

[2] C. Liu, X. Yue, K. Shi, and Z. Sun, *Spacecraft Attitude Control: A Linear Matrix Inequality Approach*, Elsevier/Science Press, 2022.

[3] Q. Meng, J. Liang, and O. Ma, "Identification of all the inertial parameters of a non-cooperative object in orbit," *Aerospace Science and Technology*, vol. 91, pp. 571–582, 2019.

[4] C. Yang, Z. Lu, Z. Yang, and K. Liang, "Parameter identification for structural dynamics based on interval analysis algorithm," *Acta Astronautica*, vol. 145, pp. 131–140, 2018.

[5] C. Wei, Y. Zhang, H. Wang, Y. Zhao, and L. Zhang, "Inertia parameter identification of space floating target during robotic exploratory grasping," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 233, no. 11, pp. 4247–4260, 2019.

[6] A. Y. Lee and J. A. Wertz, "In-flight estimation of the Cassini spacecraft's inertia tensor," *Journal of Spacecraft and Rockets*, vol. 39, no. 1, pp. 153–155, 2002.

[7] H. Chang, P. Huang, Z. Lu, Y. Zhang, Z. Meng, and Z. Liu, "Inertia parameters identification for cellular space robot through interaction," *Aerospace Science and Technology*, vol. 71, pp. 464–474, 2017.

[8] Z. Chu, Y. Ma, Y. Hou, and F. Wang, "Inertial parameter identification using contact force information for an unknown object captured by a space manipulator," *Acta Astronautica*, vol. 131, pp. 69–82, 2017.

[9] E. Wilson, C. Lages, and R. Mah, "On-line gyro-based, mass-property identification for thruster-controlled spacecraft using recursive least squares," in *The 2002 45th Midwest Symposium on Circuits and Systems*, Tulsa, OK, 2002.

[10] E. Wilson, D. W. Sutter, and R. W. Mah, "Motion-based mass- and thruster-property identification for thruster-controlled spacecraft," in *Proceedings of the 2005 AIAA Infotech@Aerospace conference*, Arlington, VA, 2005.

[11] W. C. Su, C. Y. Liu, and C. S. Huang, "Identification of instantaneous modal parameter of time-varying systems via a wavelet-based approach and its application," *Computer-Aided Civil and Infrastructure Engineering*, vol. 29, no. 4, pp. 279–298, 2014.

[12] Z. Liu, H. Fang, K. W. Wang, and J. Xu, "A parameter identification method for continuous-time nonlinear systems and its realization on a Miura-origami structure," *Mechanical Systems and Signal Processing*, vol. 108, pp. 369–386, 2018.

[13] E. Bergmann and J. Dzielski, "Spacecraft mass property identification with torque-generating control," *Journal of Guidance, Control, and Dynamics*, vol. 13, no. 1, pp. 99–103, 1990.

[14] M. Locatelli, E. Alfieri, C. H. Onder, and H. P. Geering, "Identification of the relevant parameters of the wall-wetting system by extended Kalman filtering," *Control Engineering Practice*, vol. 14, no. 3, pp. 235–241, 2006.

[15] M. C. Norman, M. A. Peck, and D. Oshaughnessy, "In-orbit estimation of inertia and momentum-actuator alignment parameters," *Journal of Guidance Control and Dynamics*, vol. 34, no. 6, pp. 1798–1814, 2011.

[16] R. Munguia, S. Urzua, and A. Grau, "EKF-based parameter identification of multi-rotor unmanned aerial vehiclesmodels," *Sensors*, vol. 19, no. 19, p. 4174, 2019.

[17] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing*, pp. 153–158, Lake Louise, AB, Canada, 2000.

[18] R. Kandepu, B. Foss, and L. Imsland, "Applying the unscented Kalman filter for nonlinear state estimation," *Journal of Process Control*, vol. 18, no. 7-8, pp. 753–768, 2008.

[19] G. Hinton, L. Deng, D. Yu et al., "Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[20] Q. Gao, J. Liu, and Z. Ju, "Robust real-time hand detection and localization for space human-robot interaction based on deep learning," *Neurocomputing*, vol. 390, pp. 198–206, 2020.

[21] H. Chen, Z. Liu, C. Alippi, B. Huang, and D. Liu, "Explainable intelligent fault diagnosis for nonlinear dynamic systems: from unsupervised to supervised learning," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2022.

[22] W. Yu, J. Tan, C. K. Liu, and G. Turk, "Preparing for the unknown: learning a universal policy with online system identification," in *Robotics: Science and Systems XIII*, Cambridge, MA, 2017.

[23] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vancouver, BC, 2017.

[24] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, QLD, Australia, 2018.

[25] Y. Dong, "An application of deep neural networks to the in-flight parameter identification for detection and characterization of aircraft icing," *Aerospace Science and Technology*, vol. 77, pp. 34–49, 2018.

[26] A. Punjani and P. Abbeel, "Deep learning helicopter dynamics models," in *2015 IEEE International Conference on Robotics and Automation*, Seattle, WA, 2015.

[27] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on Machine Learning*, Atlanta, GA, 2013.

[28] R. C. B. Rego and F. M. U. de Araujo, "Lyapunov-based continuous-time nonlinear control using deep neural network applied to underactuated systems," *Engineering Applications of Artificial Intelligence*, vol. 107, article 104519, 2022.

[29] R. Kumar, S. Srivastava, J. R. P. Gupta, and A. Mohindru, "Diagonal recurrent neural network based identification of nonlinear dynamical systems with Lyapunov stability based adaptive learning rates," *Neurocomputing*, vol. 287, pp. 102–117, 2018.

[30] R. Kumar and S. Srivastava, "Externally recurrent neural network based identification of dynamic systems using Lyapunov stability analysis," *ISA Transactions*, vol. 98, pp. 292–308, 2020.

[31] R. Kumar, "Memory recurrent Elman neural network-based identification of time-delayed nonlinear dynamical system," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no. 2, pp. 753–762, 2023.

[32] Y. H. Pao, G. H. Park, and D. J. Sobajic, "Learning and generalization characteristics of the random vector functional-link net," *Neurocomputing*, vol. 6, no. 2, pp. 163–180, 1994.

[33] G. Huang, Q. Zhu, and C. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[34] G. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 42, no. 2, pp. 513–529, 2012.

[35] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[36] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, Fort Lauderdale, USA, 2011.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE International Conference on Computer Vision*, Santiago, Chile, 2015.

[38] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *International conference on learning representations*, San Diego, CA, 2015.

[39] Y. Wu, L. Liu, J. Bae et al., "Demystifying learning rate policies for high accuracy training of deep neural networks," in *2019 IEEE International conference on big data (Big Data)*, Los Angeles, CA, USA, 2019.

[40] Q. Teng and L. Zhang, "Data driven nonlinear dynamical systems identification using multi-step CLDNN," *AIP Advances*, vol. 9, no. 8, article 085311, 2019.