Dynamic Prediction of the Thermal Nonlinear Process Based on Deep Hybrid Neural Network

Peng Wang and *Fengqi* Si[*]

Key Laboratory of Energy Thermal Conversion and Control of Ministry of Education, Southeast University, Nanjing, 210096, Jiangsu Province, China

Abstract. Nonlinear system prediction plays an important role in the practical thermal process, and deep learning algorithm is now popular in nonlinear dynamic system modeling because of its powerful learning ability. In this paper, the dynamic artificial neural networks (DANNs), which can be divided into two different types with external dynamic characteristics and internal dynamic characteristics, are analyzed. The mathematical formulations of feedforward deep neural network (DNN), traditional recurrent neural network (RNN) and Long-Short Term Memory network (LSTM) models are given. Furthermore, the structure of deep Hybrid Neural Network (DHNN) is described. Finally, the applicability of the above models in the thermal nonlinear process with different structural features is discussed. Simulation experiments reveal that DANNs with internal dynamic characteristics more suitable for solving thermal nonlinear system modeling problems with unknown order, and DHNN based on LSTM model has performed much better in approximating the dynamics of the thermal process with state parameters.

1 Introduction

Recent years, system modeling has made great progress due to the huge demand for controller design, and process analysis [1, 2]. In many cases the model can be derived based on physical knowledge about the system by simplification [3]. However, most industrial systems are non-linear, especially thermal systems, and linear models cannot be used to correctly describe the dynamic behavior of the nonlinear system. In fact, it is difficult to model nonlinear system due to uncertainty (including unknown structure and parameters) [4]. Therefore, nonlinear system dynamic prediction is a significant and challenging task in thermal nonlinear process.

System prediction is a method of identifying and predicting the dynamic characteristics of a system from measurements of the inputs and outputs. System prediction for nonlinear system usually has developed by focusing on specific classes of system and can be broadly categorized into five basic description approaches, each defined by a model class: Volterra series models [5], block structured models, neural network models [6], NARMAX models [7], State-space models [8]. The Volterra, block structured models and many neural network architectures can all be considered as subsets of the NARMAX model. Since NARMAX was introduced, by proving what class of nonlinear systems can be represented by this model, many results and algorithms have been derived based around this description. Most of the early work was based on polynomial expansions of the NARMAX model. These are still the most popular methods today, but other more complex forms based on wavelets and other expansions have been introduced to represent severely nonlinear and highly complex nonlinear systems.

Artificial neural networks (ANNs) have been widely used for nonlinear system modeling due to its powerful approximation ability. According to approximation theory, if the number of hidden neurons is sufficient and even equal to the number of training samples, a single hidden layer neural network can approximate any nonlinear system to any desired accuracy [9]. However, a neural network with the number of hidden neurons equal to training samples is impractical, which will bring huge difficulties to the training of neural network. Therefore, increasing the number of hidden layers can solve this contradictory problem to some extent.

Recently, the deep learning algorithm is now popular in nonlinear system modeling and prediction because of the strong nonlinear learning ability [10, 11]. Structurally, there are two types of neural networks: feed-forward neural network (FFNN) and RNN. In FFNN the input feeds forward through the network layers to the output and hence, only the forward connections are present between the neurons while in the case of RNN both feed-forward and feedback connections are present which makes them the nonlinear dynamic feedback systems. Only when the order of the dynamic system input and output is known or within a certain range, the data of the first n sampling moments of the input parameter and the output parameter can be added at the input layer of the FFNN, which making the overall network a nonlinear dynamic system, and the requirements for nonlinear dynamic process modeling

[*] Corresponding author: wp_edu@126.com

are met to some extent. But deducing the order of the plant is a relatively difficulty since most of the plant's dynamics are very complex and are not fully understood. Compared with FFNN's difficulty, the neural network which include the dynamics directly into its structure can learn the dynamics of the system without requiring any apriori knowledge regarding the system. These neural networks with self-feedback loops are called RNN. They have been used successfully in sequence learning tasks, such as handwriting recognition [12], speech recognition applications [13]. The LSTM which has the ability to forget and remember past hidden states belongs to the RNN category [14]. However, the deep hybrid neural network based on DNN and RNNs is rarely applied in thermal nonlinear dynamic processes modeling/ identification.

In this paper, we apply DNN, RNN, LSTM and DHNN based on LSTM models to thermal nonlinear dynamic prediction and compare the prediction ability of the above models in two situations: one is nonlinear dynamic objects with known order, the other is unknown input and output parameter order. The main outline of the paper is as follows: Section 2 gives the mathematical formulation of DNN, RNN and LSTM models. In Section 3 the Applicability of Deep learning algorithm in thermal nonlinear dynamic identification is discussed in theory. In Section 4 simulation experiments are performed by considering two types of thermal numerical examples. The performances of all three identifiers are tested and compared. The conclusion is given in Section 5.

2 Theoretical foundations of DNN, RNN and LSTM models

The memoryless steady-state artificial neural network can be extended to dynamic artificial neural networks (DANNs) by introducing dynamic elements (such as delay-operator), which can be used to predict the dynamic nonlinear characteristics of system. DANNs can be divided into two different types with external dynamic characteristics and internal dynamic characteristics.

A brief mathematical theory of the deep neural networks which can represent the above types is introduced.

2.1 DNN model

The structure of DNN which is also called Multi-Layer perceptron (MLP) can be classed into three categories: input layer, hidden layers and output layer. As shown in Fig. 1, the number of hidden layers is always more than two layers. The mathematical model of DNN is given by:

$$\begin{cases} H_1 = f_1(W_1 X + B_1) \\ H_2 = f_2(W_2 H_1 + B_2) \\ \vdots \\ H_l = f_l(W_l H_{l-1} + B_l) \\ Y = f_o(W_o H_l + B_o) \end{cases} . \tag{1}$$

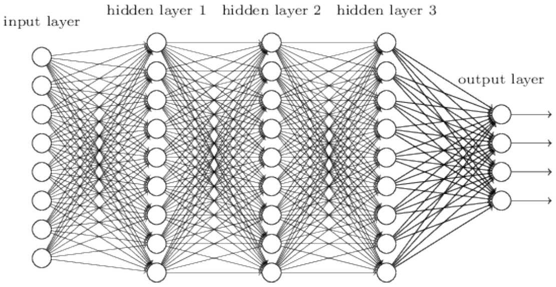

Fig. 1. Structure of DNN

Where H_l, W_l and B_l are the output, Weight matrix and bias of lth hidden layer respectively. f(·) is an activation function of each layer and nonlinear activation functions, such as: sigmoid, tanh and relu, are chosen. Y and X are the output and input of the DNN model respectively. The Back Propagation (BP) algorithm is used to calculate the weight derivatives for approximating the solution of DNN iteratively.

2.2 RNN model

Traditional RNN is a structural-improved multilayer perceptron network by introducing dynamic elements inside the hidden layer [15]. DNN can only map from input to output vector (one-to-one mapping) [16], whereas RNN can map from the entire history of previous inputs to each output (sequence-to-sequence mapping). Here a simple RNN model structure is shown in Fig.2 which include: input unit, one output unit, and one recurrent hidden unit unfolded into a full network. The forward pass of an RNN is the same as that of a multilayer perceptron with a single hidden layer except that activations arrive at the hidden layer from both the current external input and the hidden layer activations from previous timesteps. The above structure can make the RNN take advantage of the information in any long sequence in theory. The forward calculation process of RNN is given by:

$$y_t = f(Vs_{t-1} + b_o) \tag{2}$$

$$s_t = g(Ux_t + Ws_{t-1} + b_s) \tag{3}$$

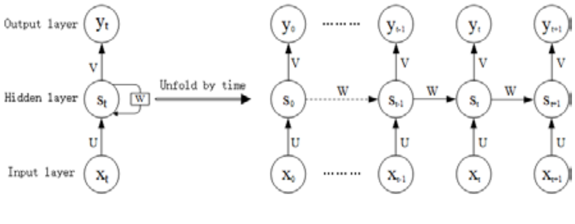

Fig. 2. Structure of RNN

In above formula, s_{t-1} is the output of hidden layer at the time of $(t-1)$. x_t is the input of hidden layer at the time of t. b_h and b_o is the bias of hidden layer and output layer. $f(\cdot)$ is map function which is usually a nonlinearity such as tanh or relu. W, U, V are the weight matrixes in different network layers. Unlike with traditional DNN, which need to use different parameters at each layer, RNN shares the same parameters (U, V, W above) across all steps which can greatly reduce the total number of train parameters [17].

A classical algorithm has been devised to efficiently calculate weight derivatives for RNN: Back Propagation Through Time (BPTT) [18]. Compared with standard BP algorithm, the performance function $E(t)$ is not only related to the current hidden layer state, but also to the hidden layer state at the previous timesteps [19]. The gradient calculation of weight matrix is written as:

$$\frac{\partial E}{\partial V}=\sum_{t=1}^{n}\frac{\partial E(t)}{\partial V} \tag{4}$$

$$\frac{\partial E}{\partial W}=\sum_{t=1}^{n}\sum_{k=0}^{t}\frac{\partial E(t)}{\partial s_t}\left(\prod_{j=k+1}^{t}\frac{\partial s_j}{\partial s_{j-1}}\right)\frac{\partial s_k}{\partial W} \tag{5}$$

$$\frac{\partial E}{\partial U}=\sum_{t=1}^{n}\sum_{k=0}^{t}\frac{\partial E(t)}{\partial s_t}\left(\prod_{j=k+1}^{t}\frac{\partial s_j}{\partial s_{j-1}}\right)\frac{\partial s_k}{\partial U} \tag{6}$$

Owing to the structure of RNN computing which adds the input of last time in the calculation process, the perception of the node in front of time node decreases as time goes by, which can be in other words, there is the gradient vanishing problem of RNN [20].

2.3 LSTM model

LSTM networks solve the problem of vanishing gradients of RNN by splitting in three inner-cell gates and build memory cells C to store information in a long-range context [21]. A typical LSTM networks cell is configured mainly by four gates: forget gate *f*, input gate i, input modulation gate $\tilde{c}$ and output gate O. Forget gate *f* decides when to forget the output results and thus selects the optimal time lag for the input sequence. Input gate i takes a new input point from outside and process newly coming data. Memory cell input gate $\tilde{c}$ takes input from the output of the LSTM networks cell in the last iteration. Output gate O takes all results calculated and generate output for the LSTM networks cell [22]. The typical structure of LSTM networks is shown in Fig. 3.

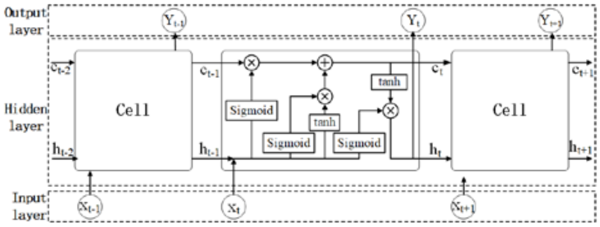

Fig. 3. Structure of LSTM

In LSTM model, a fully connected layer is applied on the output layer of the LSTM cell. Let us denote the input time series as$X=(x_1,x_2,\cdots,x_t)$, hidden state cells as $H=(h_1,h_2,\cdots,h_t)$, and output sequence as$Y=(y_1,y_2,\cdots,y_t)$. The computation of LSTM can be done as follows:

$$y_t = f(Wh_t + b_h) \tag{7}$$

$$h_t = H(h_{t-1}, c_{t-1}, x_t) \tag{8}$$

The LSTM structure depicted above is implemented through the following functions:

$$h_t = o_t \circ \tanh(c_t) \tag{9}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \tag{10}$$

$$\begin{bmatrix} f_t \\ i_t \\ \tilde{c}_t \\ o_t \end{bmatrix} = \begin{bmatrix} \sigma(W_{fh}h_{t-1}+W_{fx}x_t+b_f) \\ \sigma(W_{ih}h_{t-1}+W_{ix}x_t+b_i) \\ \tanh(W_{ch}h_{t-1}+W_{cx}x_t+b_c) \\ \sigma(W_{oh}h_{t-1}+W_{ox}x_t+b_o) \end{bmatrix} \tag{11}$$

σ and $tanh$ are applied which represent the specific, elementwise applied activation functions of the LSTM. The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. The W terms again denote weight matrices. The algorithm used to efficiently calculate weight derivatives for LSTM is Back Propagation Through Time (BPTT) which is similar as the algorithm for RNN.

3 Applicability of deep learning in thermal nonlinear system

In thermal process, establishing accurate nonlinear dynamic model is of great significance for the control and prediction of thermal processes. The dynamic behavior of thermal nonlinear process is generally represented by a NARMAX models or a State-space models.

3.1 Application of conventional deep neural network

The nonlinear autoregressive moving average model with exogenous inputs (NARMAX model) can represent a wide class of nonlinear systems [7], and is defined as

$$\begin{aligned} Y_p(k) = F[&X(k), X(k-1), \cdots, X(k-m), \\ &Y_p(k-1), Y_p(k-2), \cdots, Y_p(k-n)] \end{aligned} \tag{12}$$

where Y(k) and X(k) are the system output and input respectively; m and n are the maximum lags for the system output, input typically set to m⩽n; F[•] is some nonlinear function. The model is essentially an expansion of past inputs and outputs.

For many of times these plant's dynamics are not fully understood due to the complex of system. Deep Neural networks are known to be good approximations. They are inherently nonlinear and with the help of learning algorithms they can learn the unknown dynamics of the given plant. The DNN、RNN and LSTM based identifiers is given by:

$$\begin{aligned} Y_{DNN}(k) = \hat{F}[X(k), X(k-1), \cdots, X(k-m),& \\ Y_{DNN}(k-1), Y_{DNN}(k-2), \cdots, Y_{DNN}(k-n)]& \end{aligned} \tag{13}$$

$$Y_{RNN}(k) = \hat{F}[X(k)] \tag{14}$$

$$Y_{LSTM}(k) = \hat{F}[X(k)] \tag{15}$$

3.2 Application of deep Hybrid neural network

In a general situation, it might be the case that some exogenous uncertain disturbance passes through the nonlinear dynamics and influence the outputs. A model class that is general enough to capture this situation is the class of stochastic nonlinear state-space models. A state-space model is usually obtained using thermodynamic physical laws [8], and the parameters to be identified usually have some physical meaning or significance. A discrete-time state-space model may be defined by the difference equations:

$$\begin{aligned} S(k+1) &= F[S(k), X(k)] \\ Y(k) &= G[S(k), X(k)] \end{aligned} \tag{16}$$

Where $Y(k)$ and $X(k)$ are the system output and input respectively; k is a positive integer referring to time; The functions F and G are general nonlinear functions. The first equation is known as the state equation in which $S(k)$ is known as the state process and the second is known as the output equation.

As shown in the equation, the output of the state-space model is not only related to the state parameters, but also to the input parameters at the same time. Therefore, the DHNN based on LSTM and DNN which can correspond to the feature is proposed. The simplicial structure of DHNN is shown in Fig. 4. The mathematical model of DHNN is given by:

$$y_t = f_i(h_t, x_t) \tag{17}$$

$$h_t = H(h_{t-1}, c_{t-1}, x_t) \tag{18}$$

Where, h_t can be calculated from equation (9)-(11). The function of concatenate can used to splice h_t and x_t as the input of equation (17). $f_i(\cdot)$ is an activation function of the ith layer which has the same mathematical structure as the DNN.

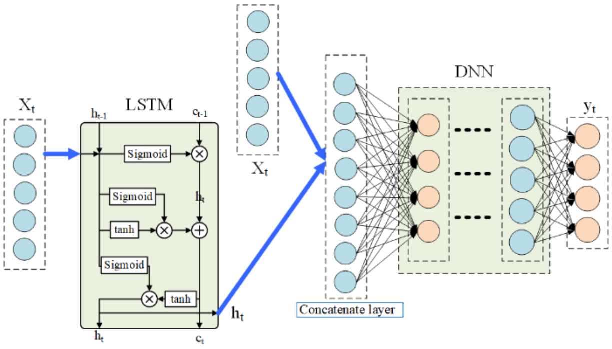

Fig. 4. Structure of DHNN

4 Simulation study

In this section, the capabilities of DHNN based identification model for nonlinear dynamic system is tested and compared with RNN and DNN based identification models through thermal process examples with two different features which are described in detail in each example. The number of neurons in the hidden layer do have an effect on the overall learning and the training time. There are number of ways/methods developed in the literature to decide upon the count of hidden neurons. In addition, two evaluated indices for the prediction performance, among which the mean square error (MSE) and the coefficient of determination pronounced R squared (R^2), are given as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{19}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \tag{20}$$

Where $\bar{y}_i$ denotes the average value of measured data.

4.1 Example 1: Identification of thermal system without internal state parameters

For some thermal systems, the nonlinear dynamic relationship between the input and the output is directly related without internal state parameter of physical meaning. Consider a nonlinear dynamical thermal system whose nonlinear relationship is assumed to be unknown and are given as in [23]:

$$\begin{aligned} y_p(k+1) = F[y_p(k), y_p(k-1), y_p(k-2),& \\ u(k), u(k-1)]& \end{aligned} \tag{21}$$

Where unknown function has the following form

$$F[x_1, x_2, x_3, x_4, x_5] = \frac{x_1 x_2 x_3 x_5 [x_3 - 1] + x_4}{[1 + x_2^2 + x_3^2]} \tag{22}$$

https://doi.org/10.1051/e3sconf/202016201007

The identification structure of DNN, RNN and LSTM are given by Eqs. (23)–(25) respectively.

$$y_{DNN}(k+1)=\hat{F}[y_{DNN}(k),y_{DNN}(k-1),y_{DNN}(k-2),u(k),u(k-1)] \quad (23)$$

$$Y_{RNN}(k+1)=\hat{F}[X(k)] \quad (24)$$

$$Y_{LSTM}(k+1)=\hat{F}[X(k)] \quad (25)$$

In this example, a total of 300 output–input training samples were generated from the plant. The training was continued for 20 epochs. During the training, the external input $r(k)$ is considered to be random having value distributed in the interval [−1, 1].

4.1.1 Discussion on the training simulation results

The responses of all identifiers at the end of the 20th epoch of the training are shown in Fig.5. It is clear from the figure that LSTM and DNN are able to capture the dynamics of the plant much better than the RNN models. This suggests better approximation capability of DNN and LSTM. The various details of this example after training are shown in Table 1.

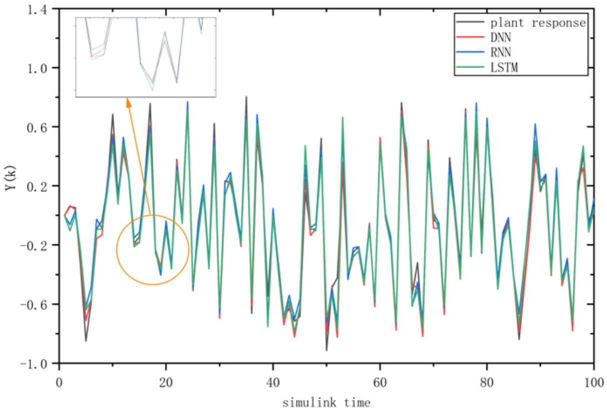

Fig. 5. Models' response at the end of training

4.1.2 Discussion on the testing simulation results

After the training, the next step is to test the performance of identifiers by using a new external input (whose values lie in a same range as that of input which was used during the training). This step is called as validation. This new external input is given by:

$$u(k)=\begin{cases}\sin(\frac{2\pi k}{250}) & \text{if } \ k\le 400\\ 0.8\sin(\frac{2\pi k}{250})+0.2\sin(\frac{2\pi k}{25}) & \text{if } \ 400\le k\le 800\end{cases} \quad (26)$$

The corresponding responses of the LSTM, RNN and DNN based identifiers are shown in Fig.6. It can be seen that LSTM response is much closer to the plant's response as compared to responses obtained with RNN and DNN identifiers. This shows that LSTM have much better ability of approximating the unknown nonlinear as compared to RNN and DNN.

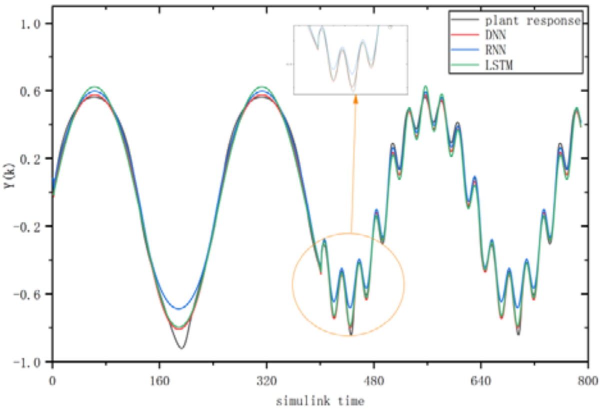

Fig. 6. Models' response used test data

Further, various details associated with this example are shown in Table 1. It can be seen from the table that minimum average MSE value is obtained with DNN and LSTM model. Also, LSTM required lesser number of parameters to be tuned compared with DNN. This makes it more computationally efficient than DNN and RNN.

Table 1. Comparison of identifiers in terms of various parameters

	DNN identifier	**RNN identifier**	**LSTM identifier**
Input neurons number	5	1	1
Structure of model	5-10-20-10-1	1-8-1	1-8-1
Total count of parameters to be tuned	501	89	329
Average MSE of training data	0.006	0.011	0.006
R^2 of training data	0.972	0.944	0.970
Average MSE of test data	0.001	0.005	0.001
R^2 of test data	0.990	0.973	0.988

4.2 Example 2: Identification of high-order thermal nonlinear system

For many large thermal systems, the nonlinear dynamic relationship is established among the state parameter, the input and the output of the system. Furthermore, some state parameters of the thermal system are unmeasurable or inaccurate. This nonlinear dynamic modeling is a big challenge for the DNN model which is essentially a memoryless neural network. Consider a nonlinear dynamical thermal system, whose dynamics are assumed to be unknown and state parameter $x(t)$ is unmeasurable, is described by the following equation:

$$y_p(k)=x(k)^2-u(k) \quad (27)$$

Here, the function of $x(t)$ and $u(t)$ has the following transfer function:

$$\frac{X(s)}{U(s)} = \frac{1}{(2s+1)^8} \tag{28}$$

For the above nonlinear dynamic system, the dynamics of state parameter $x(t)$ with $u(t)$ and $y(t)$ are unknown which means that the order of input u and output y can't be obtained. The identification structure of DNN only can be written in the following form:

$$\begin{aligned} y_{DNN}(k) = \hat{F}[&y_{DNN}(k-1), y_{DNN}(k-2), \cdots, y_{DNN}(k-m), \\ &u(k), u(k-1), \cdots, y(k-n)] \end{aligned} \tag{29}$$

Where m and n are the order which are unknown.

However, the above problem do not occur in the ANNs with internal memory structures, such as: RNN, LSTM and DHNN . Thus, the identification structure of RNN, LSTM and DHNN can be given by Eqs. (30):

$$Y_{NN}(k) = \hat{F}[X(k)] \tag{30}$$

Where *NN* is RNN, LSTM or DHNN.

For the training purpose, total of $k = 1000$ input-output samples are generated from the plant. $u(t)$ represents the input signal given by:

$$u(k) = 0.7\sin(\frac{2\pi k}{80}) + 0.3\sin(\frac{2\pi k}{140}) \tag{31}$$

In order to test the performances of these trained identification models and avoid over-fitting for the training data, we have considered the following different input signal for the validation purpose:

$$u(k) = 0.4\sin(\frac{2\pi k}{170}) + 0.1\sin(\frac{2\pi k}{60}) + 0.5\sin(\frac{2\pi k}{100}) \tag{32}$$

4.2.1 Discussion on DNN based simulation results

The simulation experiment was run with different settings of m and n which represents the order of the input and output and was run for 50 epochs. It should be noted that m should not be greater than n. The prediction performance of DNN identifiers, which was expressed by R^2, obtained at the end of the learning are shown in Fig.7. It can be seen that prediction performance of DNN identifiers increases gradually with the increase of m. The change of n has a certain impact on the prediction performance of DNN identifiers, but the impact is limited. Finally, prediction performance of DNN identifiers is not good and consumes computing resources in the case of unknown orders.

Table 2. Comparison of identifiers in terms of various parameters.

	RNN identifier	**LSTM identifier**	**DHNN identifier**
Input neurons number	1	1	1
Structure of model	1-18-1	1-9-1	1-9-18-9-1
Total count of parameters	305	406	775
Average MSE of train data	0.003	0.002	0.001
R2 of train data	0.988	0.990	0.995
Average MSE of test data	0.006	0.003	0.001
R2 of test data	0.978	0.988	0.994

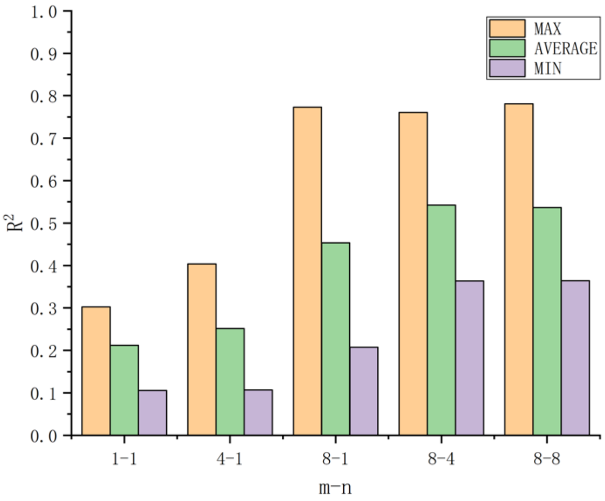

Fig. 7. R^2 of models' response used test data

4.2.2 Discussion on RNN, LSTM and DHNN based simulation results

The simulation was run for 50 epochs and in each epoch 900 input–output samples were used. The responses of the RNN, LSTM and DHNN identifiers obtained at the end of the learning are shown in Fig.8. It can be seen that the response given by all the identifiers are close to the desired trajectory but DHNN identifier response is still slightly better than the response of other identifiers. Further, the details regarding the total number of parameters to be tuned in each identifier and the MSE value obtained during the training are given in Table 2.

The validation responses of the identifier are shown in Fig. 9. It can be seen that all identifiers are following the gantry outputs corresponding to the new input. However, RNN and LSTM response is slightly Poor performance in partial dynamic process. This test shows that all recurrent identifiers were trained properly.

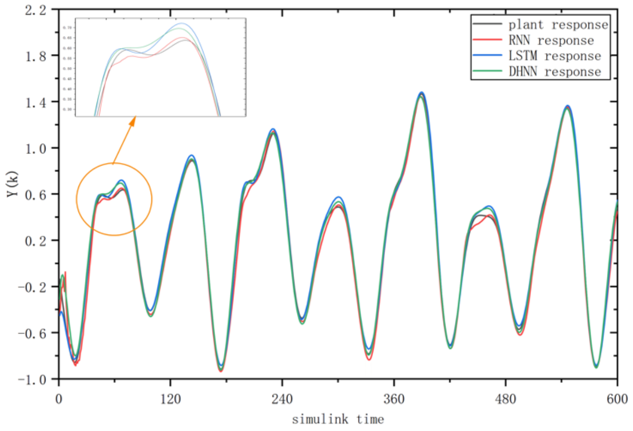

Fig. 8. Models' response at the end of training

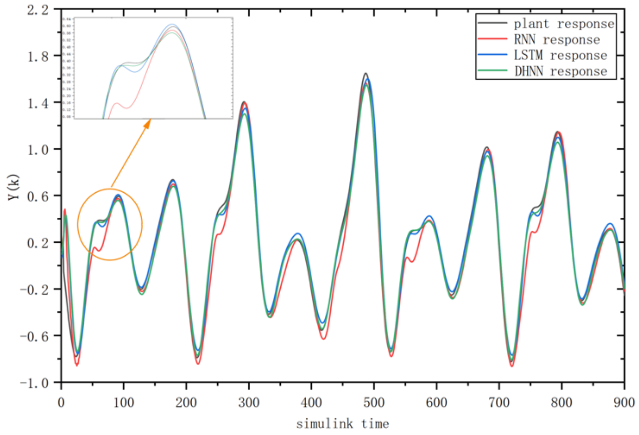

Fig. 9. Models' response used test data

5 Conclusions

In this article, the dynamic artificial neural networks applied into thermal nonlinear modeling is analyzed, and the applicability of DNN, RNN, LSTM and DHNN model mentioned in this article is discussed. The above models are test on theoretical and experimental examples which contain thermal complex nonlinear system. Simulation results show that in almost all the cases examined, the LSTM and DHNN based on LSTM has shown superior modeling accuracy and has also shown more robustness as compared to the other 2 identification models. Thus, DHNN based on LSTM can be regarded as a general identification network that can be applied to the identification of a wide class of nonlinear dynamic systems.

References

1. Jonas Martensson, and Hakan Hjalmarsson. IEEE Transactions on Automatic Control. **56**, 100 (2011)
2. Zhang, Yingwei, et al. IEEE Transactions on Neural Networks and Learning Systems. **23**,277 (2012)
3. S. Julian and J. Schoukens. Control Engineering Practice. 154 (2016)
4. Gandomi, Amir Hossein , and A. H. Alavi. Information Sciences. **181**, 5227 (2011)
5. Schetzen, Martin. (2006)
6. Lee, T. T., and J. T. Jeng. IEEE Transactions on Systems Man & Cybernetics Part B. **28**, 925(1998)
7. Ding, Feng, and T. Chen. Automatica. **41**, 1479 (2005)
8. Thomas B. Schön, A. Wills, and B. Ninness. Automatica. **47**, 39 (2011)
9. Leung, F. H. F, et al. IEEE Transactions on Neural Networks. **14**, 79 (2003)
10. Wen. Yu and D. L. R. Erick. International Journal of Machine Learning and Cybernetics. (2018)
11. Y. Kyongmin and I. Melnyk. Journal of Computational Physics. (2018)
12. Graves, A. IEEE Transactions on Pattern Analysis and Machine Intelligence. **31**, 855 (2009)
13. Graves, Alex, A. R. Mohamed, and G. Hinton. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. 6645 (2013)
14. Hochreiter, S, and J. Schmidhuber. Neural Computation **9**, 1735 (1997)
15. Lukoševičius, Mantas, and H. Jaeger. Computer Science Review. **3**, 127 (2009)
16. Doya. K. Proceedings of 1992 IEEE International Symposium on Circuits and Systems. **6**, 2777 (1992)
17. J. Faraway and C. Chatfield. Applied statistics. 231 (1998)
18. Herbert Jaeger. German National Research Center for Information Technology. (2002)
19. Yoshua Bengio, Patrice Simard, Paolo Frasconi. IEEE Transactions on Neural Networks. **5**, 157 (1994)
20. Graves, A., Jaitly, N., Mohamed, A. Automatic Speech Recognition and Understanding. (2013)
21. Felix A. Gers, Jürgen Schmidhuber, Fred A. Neural Computation. **12**, 2451 (2000)
22. S. Hochreiter and J. Schmidhuber. Neural Compute. **9**, 1735 (1997)
23. Narendra, K. S., and K. Parthasarathy. IEEE Transactions on Neural Networks. **1**, 4 (1990)