Continuous-Time State-Space Neural Network and Its Application in Modeling of Forced-Vibration Systems

HONG-WEI LI, YI-QING NI, YOU-WU WANG,
ZHENG-WEI CHEN and EN-ZE RUI

ABSTRACT

Rapid advances in machine learning make it possible to formulate surrogate models for complex forced-vibration systems using neural networks. Recently, the continuous-time state-space neural network (CSNN) has shown great potential and has been drawing growing attention from the community. In this paper, we propose a generalized CSNN model for various forced-vibration systems. The CSNN model comprises two sets of independent neural networks aimed to compute the state derivative and system response, respectively. Both neural networks adopt linear and nonlinear layers in parallel, aimed to enhance the CSNN model with the capability to recognize the linear and nonlinear behaviors of systems. Additionally, the bias options in the CSNN model are all turned off to improve the stability of the model in the long-term time-series forecast. Integration on the state derivative is executed using the explicit 4th-order Runge-Kutta method. An illustrative example is provided in this paper, demonstrating that the CSNN model can achieve high performance and training efficiency with a few hyper-parameters.

INTRODUCTION

Dynamic systems in engineering fields might be intricate and exhibit significant nonlinearities [1,2]. Traditional approaches for system identification and evaluation often require us to gain as much as we can about the physical knowledge of the system. For example, the physical model of a hydraulic actuator-specimen system should reflect the dynamics of the servo-valve, actuator, specimen, and the control-structure interaction [3–5], leading to at least a fifth-order linear or nonlinear model, depending on the complexity of the specimen [6–8]. Another example is that the memory effect of some materials could be well interpreted by the fractional derivative order model, while solving it in the time domain is challenging [9–13]. Such physical models need to be carefully dealt with using sophisticated mathematical tools and thereby are not attractive to engineers.

The research community has gained great progress in leveraging neural networks to identify or represent forced-vibration systems. For example, the state-space neural

Hong-Wei Li, Yi-Qing Ni, You-Wu Wang, Zheng-Wei Chen, En-Ze Rui. Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University; National Rail Transit Electrification and Automation Engineering Technology Research Center (Hong Kong Branch), Hung Hom, Kowloon, Hong Kong, China

network has been developed to represent forced-vibration systems in the discrete-time domain [14, 15]. Long short-term memory (LSTM) neural network has been adopted to predict the dynamic responses of nonlinear systems [16, 17]. Convolutional neural network (CNN) has been used for structural black-box modeling, damage detection, and loss data reconstruction [18, 19]. Additionally, by adding essential physical information of systems to LSTM and CNN, the enhanced neural networks PhyLSTM [20] and PhyCNN [21] have been formed for structural seismic response modeling. In the above studies, the neural networks only function at a particular sampling rate after they are trained because they were all constructed in the discrete-time domain. Recently, neural ordinary differential equations (NODE) for the formulation of continuous-time neural network models have been widely investigated [22, 23]. The NODE method works regardless of the change of sampling rate and has exhibited high performance in learning the unmodeled nonlinear dynamics of systems [24, 25]. Furthermore, the continuous-time state-space neural network (CSNN) has been developed based on the idea of the NODE method and has shown improved performance and efficiency in the identification of nonlinear systems [26, 27], therefore has enormous potential for applications in modeling and response prediction of dynamic systems. How to construct CSNN models for various forced-vibration systems is an appealing research topic and deserves further investigation.

In this paper, we establish a generalized CSNN model for forced-vibration systems in civil engineering. The state vector is introduced in the CSNN model as the hidden variable. The CSNN model consists of two independent sets of neural networks, labeled as state and output calculators, which are used to compute the state derivative and output vectors, respectively. Both the state and output calculators adopt linear and nonlinear neural network layers in parallel, enabling the CSNN model to capture the linear and nonlinear components in the responses of the system. The integration operations from state derivative to state using the explicit 4th-order Runge-Kutta (RK4) method make the CSNN model independent of the data sampling rate. Additionally, the CSNN model does not require the input data length to be fixed. With the above features, the CSNN model is highly flexible and has a strong capability to predict system responses in real time.

This paper is organized as follows. First, the CSNN modeling methodology is presented, and the features and potential applications of the CSNN model are discussed in detail. Then, the performance of the CSNN model is evaluated through a numerical nonlinear example. Last, some conclusions are drawn.

CONTINUOUS-TIME STATE-SPACE NEURAL NETWORK (CSNN)

In this study, the forced-vibration system with excitations (*input vector*): $\mathbf{u}(t) \in \mathbb{R}^{n_u \times 1}$ and responses (*output vector*): $\mathbf{y}(t) \in \mathbb{R}^{n_y \times 1}$ is studied. Assuming that the true physical model of the system is completely unknown (black box), we will utilize the CSNN architecture to formulate a surrogate model of the system. The dynamic equation of an unknown system at any time t is constructed using a CSNN cell as shown in Figure 1, given by

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathcal{N}_x \left[\mathbf{x}(t), \mathbf{u}(t)\right], \\ \hat{\mathbf{y}}(t) &= \mathcal{N}_y \left[\mathbf{x}(t), \mathbf{u}(t)\right]. \end{aligned} \tag{1}$$

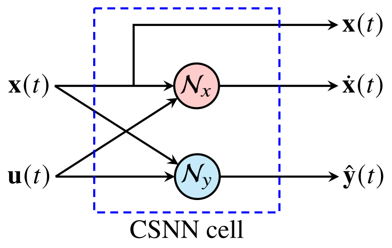

Figure 1. Structure of the CSNN cell at time t.

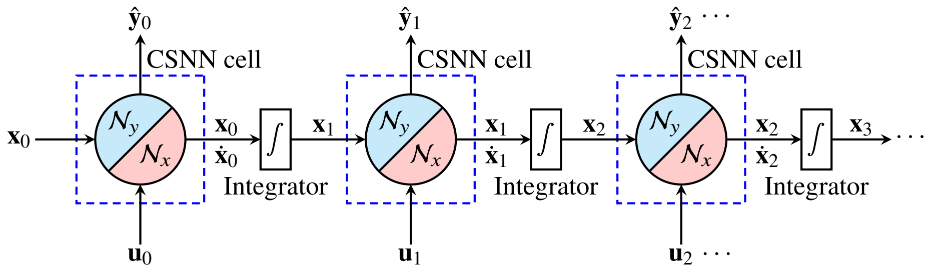

Figure 2. Illustration of CSNN for time-series modeling of a forced-vibration system.

In Equation (1), $\mathbf{x}(t) \in \mathbb{R}^{n_x \times 1}$ is the *state vector* which is used to learn the system behavior along the vibration trajectory; $\dot{\mathbf{x}}(t)$ is the *state derivative vector*; $\hat{\mathbf{y}}(t) \in \mathbb{R}^{n_y \times 1}$ is the *predicted output vector*; $\mathcal{N}_x(\cdot)$ and $\mathcal{N}_y(\cdot)$ are two neural networks, defined as the *state calculator* and *output calculator*, respectively. The initial state vector is assumed to be $\mathbf{0}$.

Figure 2 provides an illustration of CSNN for time-series modeling of a forced-vibration system, where

$$\mathbf{x}_i = \mathbf{x}(t_i),\ \mathbf{u}_i = \mathbf{u}(t_i),\ \hat{\mathbf{y}}_i = \hat{\mathbf{y}}(t_i),\ i = 0,\ 1,\ 2, ..., \tag{2}$$

$t_0 = 0\,\text{s}$, and $[t_0,\ t_1,\ t_2, ...]$ is an ascending time series. The time interval between t_i and t_{i+1} is defined as $\Delta t_i = t_{i+1} - t_i$, which might vary with time. All the CSNN cells shown in Figure 2 share the same model parameters.

The operations and neural network layers used in the state and output calculators are described in Figure 3. For both the state and output calculators, the state and input vectors are concatenated first; then fed into a sequence of fully connected nonlinear layers and a single fully connected linear layer to calculate nonlinear and linear results, respectively; and finally, the nonlinear and linear results are summed up to generate the state derivative vector or output vector we need.

The graph representations of the state and output calculators are shown in Figure 4, where the number of nonlinear layers in the state and output calculators is p and q, respectively. The hyperbolic tangent function, i.e., Tanh$(\cdot)$, is adopted as the activation

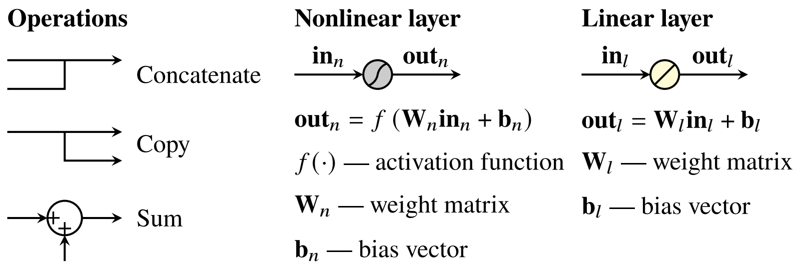
Operations
Concatenate
Copy
Sum
Nonlinear layer
$\mathbf{in}_n$
$\mathbf{out}_n$
$\mathbf{out}_n = f\,(\mathbf{W}_n \mathbf{in}_n + \mathbf{b}_n)$
$f(\cdot)$ — activation function
$\mathbf{W}_n$ — weight matrix
$\mathbf{b}_n$ — bias vector
Linear layer
$\mathbf{in}_l$
$\mathbf{out}_l$
$\mathbf{out}_l = \mathbf{W}_l \mathbf{in}_l + \mathbf{b}_l$
$\mathbf{W}_l$ — weight matrix
$\mathbf{b}_l$ — bias vector

Figure 3. Operations and layers used in the state and output calculators.

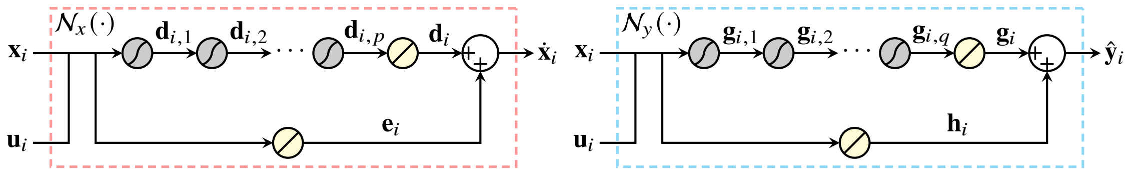

Figure 4. Graph representations of the state and output calculators (left: state calculator; right: output calculator).

function in each nonlinear layer. The last nonlinear layer is followed by a fully connected linear layer to make the output vary in the entire real-number domain. N sets of observed input and output vectors $[(\mathbf{u}_0, \mathbf{y}_0), (\mathbf{u}_1, \mathbf{y}_1), ..., (\mathbf{u}_{N-1}, \mathbf{y}_{N-1})]$ are used to train the CSNN model, i.e., to minimize the errors between the predictions $[\hat{\mathbf{y}}_0, \hat{\mathbf{y}}_1, ..., \hat{\mathbf{y}}_{N-1}]$ and observations $[\mathbf{y}_0, \mathbf{y}_1, ..., \mathbf{y}_{N-1}]$.

The RK4 method is used in this paper for the integration of the state vector. The neural network $\mathcal{N}_x(\cdot)$ is treated as a general function in implementing the RK4 method. In addition to the RK4 method, the Euler method is another possible option. Nevertheless, the Euler method is not recommended due to its low precision, particularly when using relatively large time intervals. Adam algorithm is adopted to optimize the model parameters during the training process in this study, and alternative algorithms such as stochastic gradient descent (SGD) algorithm and L-BFGS algorithm can also be adopted. Furthermore, multiple independent input-output datasets (batched data) can be considered in the training and prediction processes of the CSNN model.

The effectiveness of a trained CSNN model is not influenced by different data sampling rates/time intervals. And the CSNN model could be formed as a single-time-step moving-forward model, with one input and one output (one-to-one) at each time point. Consequently, the CSNN model can be used to predict the system responses in real time regardless of the input data length. Additionally, compared to existing CSNN models reported in the literature, the CSNN model proposed in this paper has the following modifications which are made specifically for the modeling of forced-vibration systems in civil engineering:

1. The hyper-parameters of existing CSNN models included both weights and biases, while all the bias options in the state and output calculators of the proposed CSNN model are turned off. This modification is based on the fact that almost all forced-vibration systems in civil engineering are dissipative systems without drift. Turning off the bias options is a straightforward way to satisfy this requirement. The authors have tried to train the CSNN models with the bias options turned off for the illustrative examples that will be presented later and found that better performance and faster training speed were achieved compared to the situation where the bias options were turned on.
2. In literature, the state vectors of CSNN models normally had physical meanings, and their initial values were carefully handled. However, we do not assign physical meanings to the state vector of the proposed CSNN model, and the state vector is merely treated as the hidden variable to process data-driven modeling of forced-vibration systems. To guarantee that zero input series generate zero output series, the initial state vector is assumed to be zero in this paper.
3. Nonlinear neural network layers were usually used alone in previous studies since

they are very powerful at fitting the training datasets, and it appeared unnecessary to adopt linear layers. However, this strategy normally results in a large number of hyper-parameters and might be inefficient for time-series problems. Instead of using nonlinear layers alone, the proposed CSNN model adopts both linear and nonlinear neural network layers, enabling the model to capture and balance linear and nonlinear components in the system responses.

The features listed above make the CSNN model workable for various problems within the domain of civil engineering such as system modeling, identification, and structural health monitoring.

ILLUSTRATIVE EXAMPLE

A single degree-of-freedom (DOF) nonlinear system subjected to acceleration excitations is investigated in this illustrative example. The nonlinear system is expressed as follows:

$$m\ddot{y}(t) + c\dot{y}(t) + k_1 y(t) + k_2 y^3(t) = -m\ddot{u}_g(t), \tag{3}$$

where the parameter values are mass $m = 1\,\mathrm{kg}$, damping coefficient $c = 1\,\mathrm{Ns/m}$, linear stiffness coefficient $k_1 = 20\,\mathrm{N/m}$, and nonlinear stiffness coefficient $k_2 = 200\,\mathrm{N/m^3}$ [21]. The database contains 99 independent seismic acceleration histories ($\ddot{u}_g(t)$) and the corresponding system displacement responses ($y(t)$). The first 10 datasets labeled as cases 1-10 are used for training and the rest (cases 11-99) are used for testing. Each input sequence is sampled at 20 Hz ($\Delta t = 0.05\,\mathrm{s}$) with a time duration of 50 seconds, leading to 1,001 data points. The PhyCNN model has the following limitations. It is executed at the specific time interval $\Delta t = 0.05\,\mathrm{s}$ and does not work for other time intervals. Additionally, it takes all 1,001 acceleration data points in 50 seconds as the single input vector to calculate the 1,001-point system responses. The input sequence with a different time duration must be cut off or padded to be a sequence of 50 seconds. The proposed CSNN model is not subject to these limitations. It is a one-to-one neural network model, and a well-trained CSNN model can be used to predict the system response with different time intervals and time durations.

The PhyCNN model consists of 5 convolution layers, 5 rectified linear unit (ReLU) activation functions (following at the end of each convolution layer), and finally, 3 fully connected linear layers. Each convolution layer has 64 filters and 50 kernels. The parameter configuration of the CSNN model are described as follows. Two hidden states ($n_x = 2$) are used. The state calculator $\mathcal{N}_x(\cdot)$ includes two nonlinear layers ($p = 2$) with two neurons ($n_d = 2$) in each layer, and the output calculator $\mathcal{N}_y(\cdot)$ includes one nonlinear layer ($q = 1$) with one neuron ($n_g = 1$). As a result, the CSNN model has significantly fewer parameters than the PhyCNN model.

The Pearson corr. between the ground truth and model prediction of the system response for the 99 cases are displayed in Figure 5. The PhyCNN model fits the training cases well except for case 8 and case 10. However, The PhyCNN model exhibits inconsistent performance for the testing cases, where the minimum Pearson corr. is 0.57781 (case 94). In contrast, the CSNN model achieves good performance for both training and testing cases. Therefore, the CSNN model is more stable than the PhyCNN model for this nonlinear illustrative example. The ground truth and model prediction of the displacement responses for cases 30 and 94 are shown in Figure 6. It is apparent from

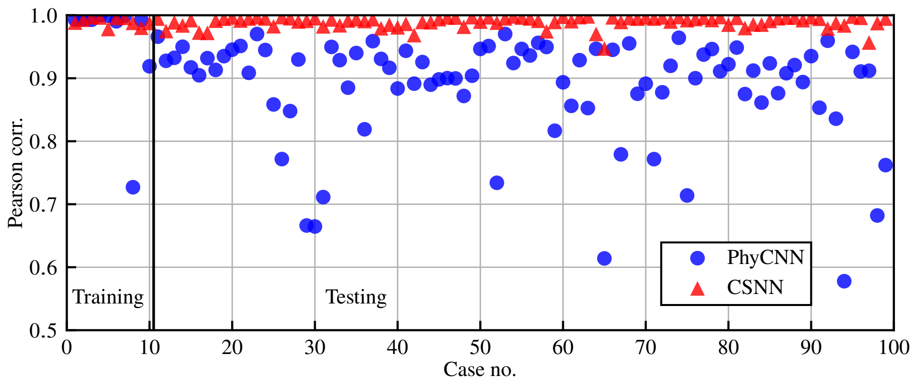

Figure 5. Pearson corr. between the ground truth and model prediction of the system response for the 99 cases of illustrative example 2.

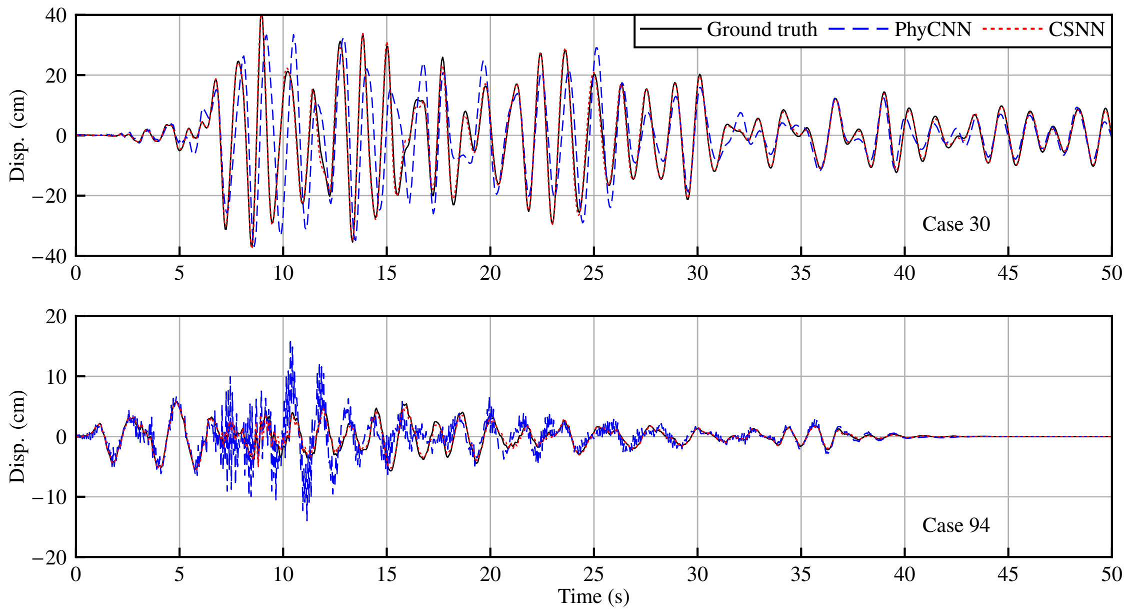

Figure 6. Ground truth and model prediction of the displacement responses for cases 29, 30, 65, and 94 of illustrative example 2.

Figure 6 that the CSNN model predicts the system responses better than the PhyCNN model.

CONCLUDING REMARKS

This paper presented the CSNN architecture to elicit data-driven surrogate models for forced-vibration systems. The CSNN model includes state and output calculators, which are used to compute the state derivative and output vectors, respectively. Both state and output calculators consist of several nonlinear neural network layers and a linear neural network layer in parallel. The RK4 method is utilized to update the state vector and calculate the output vector. The execution of the CSNN model does not require the time interval and time duration of the input data to be fixed, therefore is highly flexible and adaptive.

An Illustrative example compares the CSNN model with the PhyCNN model for the response prediction of a highly nonlinear numerical system. The results show that the CSNN model achieves better and more stable performance than the PhyCNN model. Based on the current work, we believe that the proposed CSNN model is highly promising for engineering practice.

ACKNOWLEDGMENT

The work described in this paper was supported by a grant from the Research Grants Council (RGC) of the Hong Kong Special Administrative Region (SAR), China (Grant No. PolyU 152308/22E). The authors also appreciate the funding support by the Innovation and Technology Commission of Hong Kong SAR Government to the Hong Kong Branch of National Engineering Research Center on Rail Transit Electrification and Automation (Grant No. K-BBY1) and The Hong Kong Polytechnic University's Postdoc Matching Fund Scheme (Grant No. 1-W24B).

REFERENCES

1. Khalil, H. 2002. *Nonlinear Systems*, Prentice Hall, Upper Saddle River, New Jersey, 3rd edn., ISBN 9780130673893.
2. Li, H.-W., F. Wang, Y.-Q. Ni, Y.-W. Wang, and Z.-D. Xu. 2022. "An Adaptive and Robust Control Strategy for Real-Time Hybrid Simulation," *Sensors*, 22(17):6569.
3. Dyke, S., B. Spencer Jr, P. Quast, and M. Sain. 1995. "Role of control-structure interaction in protective system design," *Journal of Engineering Mechanics*, 121(2):322–338.
4. Stehman, M. and N. Nakata. 2016. "IIR compensation in real-time hybrid simulation using shake tables with complex control-structure-interaction," *Journal of Earthquake Engineering*, 20(4):633–653.
5. Li, H.-W., A. Maghareh, H. Montoya, J. W. C. Uribe, S. J. Dyke, and Z.-D. Xu. 2021. "Sliding mode control design for the benchmark problem in real-time hybrid simulation," *Mechanical Systems and Signal Processing*, 151:107364.
6. Maghareh, A., C. E. Silva, and S. J. Dyke. 2018. "Parametric model of servo-hydraulic actuator coupled with a nonlinear system: experimental validation," *Mechanical Systems and Signal Processing*, 104:663–672.
7. Silva, C. E., D. Gomez, A. Maghareh, S. J. Dyke, and B. F. Spencer Jr. 2020. "Benchmark control problem for real-time hybrid simulation," *Mechanical Systems and Signal Processing*, 135:106381.
8. Li, H.-W., A. Maghareh, J. Wilfredo Condori Uribe, H. Montoya, S. J. Dyke, and Z. Xu. 2022. "An adaptive sliding mode control system and its application to real-time hybrid simulation," *Structural Control and Health Monitoring*, 29(1):e2851.
9. Monje, C. A., Y.-Q. Chen, B. M. Vinagre, D.-Y. Xue, and V. Feliu-Batlle. 2014. *Fractional-Order Systems and Controls: fundamentals and Applications*, London: Springer.
10. Li, H.-W., D. Gomez, S. J. Dyke, and Z.-D. Xu. 2020. "Fractional differential equation bearing models for base-isolated buildings: framework development," *Journal of Structural Engineering*, 146(2):04019197.
11. Li, H.-W., D. Gomez, S. J. Dyke, Z.-D. Xu, and J. Dai. 2021. "Investigating Coupled Train-Bridge-Bearing System Under Earthquake-and Train-Induced Excitations," *Journal of Vibration and Acoustics*, 143(5).
12. Li, H.-W., Z.-D. Xu, D. Gomez, P.-P. Gai, F. Wang, and S. J. Dyke. 2022. "A Modified Fractional-Order Derivative Zener Model for Rubber-Like Devices for Structural Control," *Journal of Engineering Mechanics*, 148(1):04021119.

13. Li, H.-W., Z.-D. Xu, F. Wang, P.-P. Gai, D. Gomez, and S. J. Dyke. 2023. “Development and Validation of a Nonlinear Model to Describe the Tension–Compression Behavior of Rubber-Like Base Isolators,” *Journal of Engineering Mechanics*, 149(2):04022104.
14. Rangapuram, S. S., M. W. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski. 2018. “Deep State Space Models for Time Series Forecasting,” in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., vol. 31, pp. 1–10.
15. Pulido, B., J. M. Zamarreño, A. Merino, and A. Bregon. 2019. “State space neural networks and model-decomposition methods for fault diagnosis of complex industrial systems,” *Engineering Applications of Artificial Intelligence*, 79:67–86.
16. Zhang, R., Z. Chen, S. Chen, J. Zheng, O. Büyüköztürk, and H. Sun. 2019. “Deep long short-term memory networks for nonlinear structural seismic response prediction,” *Computers & Structures*, 220:55–68.
17. Yue, Z., Y. Ding, H. Zhao, and Z. Wang. 2022. “Mechanics-guided optimization of an LSTM network for real-time modeling of temperature-induced deflection of a cable-stayed bridge,” *Engineering Structures*, 252:113619.
18. Ghiasi, A., M. K. Moghaddam, C.-T. Ng, A. H. Sheikh, and J. Q. Shi. 2022. “Damage classification of in-service steel railway bridges using a novel vibration-based convolutional neural network,” *Engineering Structures*, 264:114474.
19. Chen, L., W. Chen, L. Wang, C. Zhai, X. Hu, L. Sun, Y. Tian, X. Huang, and L. Jiang. 2023. “Convolutional neural networks (CNNs)-based multi-category damage detection and recognition of high-speed rail (HSR) reinforced concrete (RC) bridges using test images,” *Engineering Structures*, 276:115306.
20. Zhang, R., Y. Liu, and H. Sun. 2020. “Physics-informed multi-LSTM networks for metamodeling of nonlinear structures,” *Computer Methods in Applied Mechanics and Engineering*, 369:113226.
21. Zhang, R., Y. Liu, and H. Sun. 2020. “Physics-guided convolutional neural network (PhyCNN) for data-driven seismic response modeling,” *Engineering Structures*, 215:110704.
22. Chen, R. T. Q., Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. 2018. “Neural Ordinary Differential Equations,” in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., vol. 31, pp. 1–13.
23. Massaroli, S., M. Poli, J. Park, A. Yamashita, and H. Asama. 2020. “Dissecting Neural ODEs,” in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., vol. 33, pp. 3952–3963.
24. Lai, Z., C. Mylonas, S. Nagarajaiah, and E. Chatzi. 2021. “Structural identification with physics-informed neural ordinary differential equations,” *Journal of Sound and Vibration*, 508:116196.
25. Rahman, A., J. Drgoňa, A. Tuor, and J. Strube. 2022. “Neural Ordinary Differential Equations for Nonlinear System Identification,” *arXiv preprint arXiv:2203.00120*:1–6, doi:10.48550/ARXIV.2203.00120.
26. Forgione, M. and D. Piga. 2021. “Continuous-time system identification with neural networks: model structures and fitting criteria,” *European Journal of Control*, 59:69–81.
27. Beintema, G. I., M. Schoukens, and R. Tóth. 2023. “Continuous-time identification of dynamic state-space models by deep subspace encoding,” in *The Eleventh International Conference on Learning Representations*, pp. 1–15.