Applied Intelligence (2021) 52:11160–11171
https://doi.org/10.1007/s10489-021-02902-5

Data-driven physical law learning model for chaotic robot dynamics prediction

Kui Qian[1] · Lei Tian[1]

Accepted: 5 October 2021 / Published online: 21 January 2022

Abstract
A robot control system is a multivariable, nonlinear automatic control system, as well as a dynamic coupling system. To address the difficult problem of data prediction under a chaotic system, a data-driven physical law learning model (DPM) is proposed, which can learn the underlying physical rules that the data follow. First, two independent autoencoder neural networks are stacked and merged to explore potential physical rules. Then, a virtual Hamiltonian represented as the sum of kinetic energy and potential energy of chaotic data is introduced. Combined with the Hamiltonian equation, the learned Hamiltonian is transformed into a symplectic transformation, whose first-order differential w.r.t. the generalized coordinates and momentum can be regarded as a time-dependent prediction instead of a direct numerical approximation. Finally, the DPM continuously learns implicit Hamiltonian equations from chaotic data until it can fit the law of phase space motion in a chaotic environment. The experimental results show that the model has a better robot dynamics prediction ability in long-term chaotic systems than the existing SOTA methods.

Keywords Robot control · Data-driven · Autoencoder neural network · Hamiltonian · Dynamics prediction

1 Introduction

Data prediction under chaotic systems is a difficult task. Robot dynamics is a complex dynamic system [1–3], and the dynamic response of task execution depends on the robot dynamics model and the control algorithm. With the continuous development of robot technology, robot dynamics prediction has been widely used in various fields [4, 5] and it plays an important role in robot design and finalization, high-performance trajectory tracking during operation [6], load recognition [7], collision detection [8], and predictive maintenance [9]. Generally, a 4×4 homogeneous transformation matrix and a 3×3 rotation transformation matrix are used, starting from the position of each joint of the robot and applying certain mechanical principles to obtain the robot dynamics model [10].

Robot dynamics clearly describes the relationship between robot force and motion [11]. For a long time, classical physics laws such as the Newton-Euler and Lagrangian methods have been used for dynamic modeling [12, 13] to realize the derivation of generalized coordinates and generalized velocity in the form of dynamic equations. In order to fundamentally reveal the laws that must be followed by the variables causing the motion, these models are relatively complex, and the computational cost increases rapidly with the increase of the freedom of the robot [14, 15]. Fortunately, with the recent development of artificial intelligence technology [16–18] represented by fuzzy control and neural networks, the research field of robot dynamics models has undergone great changes in recent years [19–23]. Model-free approaches that attempt to learn the underlying regulatory rules without relying on any model assumptions have been proposed [24–26]. Neural networks have powerful data fitting and prediction capabilities and do not need various labor-intensive decompositions or transformations of complex robot dynamics equations [27–29].

A conventional neural network simulates natural human neurons by adjusting the weights and deviations of a large amount of data, thereby minimizing the difference between the actual output and the expected value. Most of the research work focuses on neural network structures and their variants for robot applications. However, the multiple variables that the dynamic system depends on, such as energy, momentum, and angular momentum,

✉ Kui Qian
kuiqian@njit.edu.cn

1 School of Automation, Nanjing Institute of Technology, Nanjing, China

all obey certain laws of physics. Conventional neural network methods usually violate these conservation laws, propagating errors in time, which renders them of limited use in complex high-dimensional real-world environments comprised of many degrees of freedom. In other words, conventional neural networks do not understand or learn the laws of physics.

Teaching physics to neural networks can make these networks adapt better to chaos in their environment, which will improve the predictive performance of conservative systems [30–33]. Researchers have found that incorporating the symplectic phase space structure of Hamiltonian dynamics is valuable [34–37]. Greydanus [38] introduced the Hamiltonian function in physics into an artificial intelligence neural network so that the neural network could make corresponding adjustments in the face of the chaos in the system and realize end-to-end unsupervised physical quantity learning. Bertalan [39] extracted phase space coordinates and a Hamiltonian function of them and used a neural network component to estimate the transformation from observations to the phase space of a Hamiltonian system. Scott [40] trained Hamiltonian neural networks (HNNs) on chaotic dynamic systems to improve their learning and prediction capabilities. These new theoretical methods have opened up new directions in the field of chaotic data analysis applications. An overall comparison between the conventional neural network and the data-driven learning of physical parameter model architecture is shown in Fig. 1.

A robot dynamics system is a typical example of a chaotic system. To solve the chaotic state response distortion problem of the traditional neural network in the robot dynamics prediction system, a data-driven physical law learning model (DPM) is proposed. First, two independent autoencoder neural networks are stacked and merged to explore the potential physical rules. Then, a virtual first-order differentiable Hamiltonian is introduced, and the sum of kinetic energy and the potential energy of chaotic data is also introduced. Combined with the Hamiltonian equation, the learned Hamiltonian is transformed into a symplectic transformation so that the derivative of the physical state w.r.t. time can be converted into the derivative of the Hamiltonian w.r.t. the generalized coordinates and the momentum. Finally, the DPM continuously learns implicit Hamiltonian equations from chaotic data until it can fit the law of phase space motion in a chaotic environment. The experimental results show that the model has better robot dynamics prediction ability in long-term chaotic systems than the existing SOTA methods. The main contributions of this paper are as follows:

(1) To overcome the chaotic blindness problem of dynamic system prediction using conventional neural network methods.
(2) To improve the network structure to enhance the learning ability of potential Hamiltonian physical quantities H.
(3) To learn multidimensional physical quantities and predict time-related dynamic physical quantities with time-independent mathematical calculations.
(4) To achieve a better understanding of the underlying dynamics than the existing SOTA methods.

2 Hamiltonian neural network

2.1 Hamilton system

Any smooth real-valued function H on a symplectic manifold can be used to define a Hamiltonian system. The function H is called the Hamiltonian or energy function. The symplectic manifold is called the phase space. Contrary to the Lagrangian quantity L, which is the difference between the kinetic energy T and the potential energy V, the Hamiltonian function H is the sum of the kinetic energy T and the potential energy V, that is, $H = T + V$. Assuming that the Lagrangian quantity of a physical system is L, the

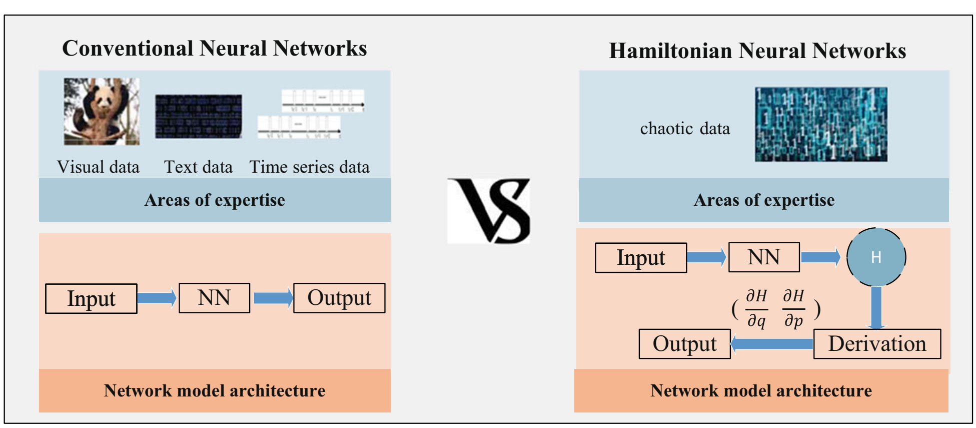


Fig. 1 Comparison of schematic architectures of different dynamic neural networks

motion of this physical system is expressed as the following Lagrangian equation:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial \dot{q}_i} = 0 \tag{1}$$

where t is the time, $q = q_1, q_2, ...q_n$ is the generalized coordinate, and $q = \dot{q}_1, \dot{q}_2, ...\dot{q}_n$ is the generalized velocity. Then,

$$dL = \sum_i \frac{\partial L}{\partial q_i} dq_i + \sum_i \frac{\partial L}{\partial \dot{q}_i} d\dot{q}_i + \frac{\partial L}{\partial t} dt \tag{2}$$

Introducing generalized momentum $p = p_1, p_2, ...p_n$, and $p_i = \frac{\partial L}{\partial \dot{q}_i}$ Then the formula is shown as follows:

$$\sum_i \dot{q}_i dp_i - \sum_i \dot{p}_i dq_i - \frac{\partial L}{\partial d} dt = d\left(\sum_i \frac{\partial L}{\partial \dot{q}_i}\dot{q}_i - L\right) \tag{3}$$

The right side of the above equation is the Hamiltonian, denoted as

$$H(q_i, p_i, t) \overset{\text{def}}{=} \sum_i \frac{\partial L}{\partial \dot{q}_i}\dot{q}_i - L \tag{4}$$

The following can be obtained:

$$dH(q_i, p_i, t) = \sum_i \dot{q}_i dp - \sum_i \dot{p}_i dq - \frac{\partial L}{\partial t} dt \tag{5}$$

Thus, the Hamiltonian canonical equation is obtained:

$$\dot{q}_i = \frac{\partial H}{\partial p_i}, \dot{p}_i = -\frac{\partial H}{\partial q_i} \tag{6}$$

For dynamic systems, although time is continuous, it can be discretely processed from the initial state to the final state of the system through a set of differential equations:

$$(q_1, p_1) = (q_0, p_0) + \int_{t_0}^{t_1} \left(\frac{\partial H}{\partial p}, -\frac{\partial H}{\partial q}\right) dt \tag{7}$$

The Hamiltonian function explains the complete information of a dynamic physical system, including kinetic energy, potential energy and the total amount of all energies. A neural network that understands Hamiltonian law will be able to learn more dynamic information.

2.2 The DPM

In robot dynamics prediction based on a conventional neural network model, the input is generally the robot's generalized coordinates and the generalized momentum (q, p), and the output of the model is $(\dot{q}, \dot{p})$. The output has a strong physical meaning and can accurately describe the relationship between the robot's motion state and force, which is of great significance to the control system. However, a conventional neural network only uses numerical calculation methods to continuously approximate and does not consider the underlying physical laws contained in the numerical value. A Hamiltonian neural network is an end-to-end learning network model that uses the powerful generalization ability of neural networks to learn potential Hamiltonian function parameters from data.

To obtain $\left(\frac{\partial H}{\partial p}, -\frac{\partial H}{\partial q}\right)$, the DPM is used to learn the parameters of the H function. The overall architecture of the DPM is shown in Fig. 2.

The network consists of two independent autoencoder networks through stacked modules and fusion modules. U_{1-8} and D_{1-8} are fully connected modules, and the number of output neurons in the last layer is the number of types of the input physical quantities. For the robot dynamics prediction model, the inputs are generalized coordinates and generalized momentum based on observable variables. The model no longer directly outputs the dynamic predictions, yet the hidden output of the model can be seen as a virtual Hamiltonian, which is first-order differentiable [41].

$$H(z) = \mathcal{F}(a_1(z), a_2(z)) \tag{8}$$

where $a_i(i = 0, 1)$ is the nonlinear transformation of the i-th layer self-encoding network and $\mathcal{F}$ is the nonlinear transformation of the multilayer stacked network. The virtual Hamiltonian passes through a differentiator to perform derivative operations on q_k and p_k and then obtains the model output value through symplectic transformation, as shown in the ST module in Fig. 2. Thus, the derivative of the input w.r.t. time can be converted into the derivative of the Hamiltonian w.r.t. the input, and the neural network can learn the implicit Hamiltonian rules through the backward propagation learning algorithm to approximate the data during the observation time. The symplectic transformation is shown in Algorithm 1:

Algorithm 1 Symplectic transformation algorithm.

Require: Model input $q_1, q_2, ..., q_k$ and $p_1, p_2, ..., p_k, n = len(input)$
Require: Hamiltonian value H
1: **while** not reach the stop training condition **do**
2: Calculate the derivative of H w.r.t. the input, and get $H_d = \left(\frac{\partial H}{\partial q_k}, \frac{\partial H}{\partial p_k}\right)$
3: Initializes the n-order identity matrix I_n
4: Perform matrix transformation on I_n to get $S_n = [-I_n[n//2 :], I_n[: n//2]]$
5: Complete symplectic transformation $H_s = H_d \times S_n$
6: **end while**

Let $z = (q_1, ..q_k, p_1, ...p_k)^T \in \Re^D$ and S be the $D \times D$ transformation matrix shown in Algorithm 1. Then, the Hamiltonian equations can be written in compact vector form:

$$\dot{z} = S \cdot \nabla_z H(z) \tag{9}$$

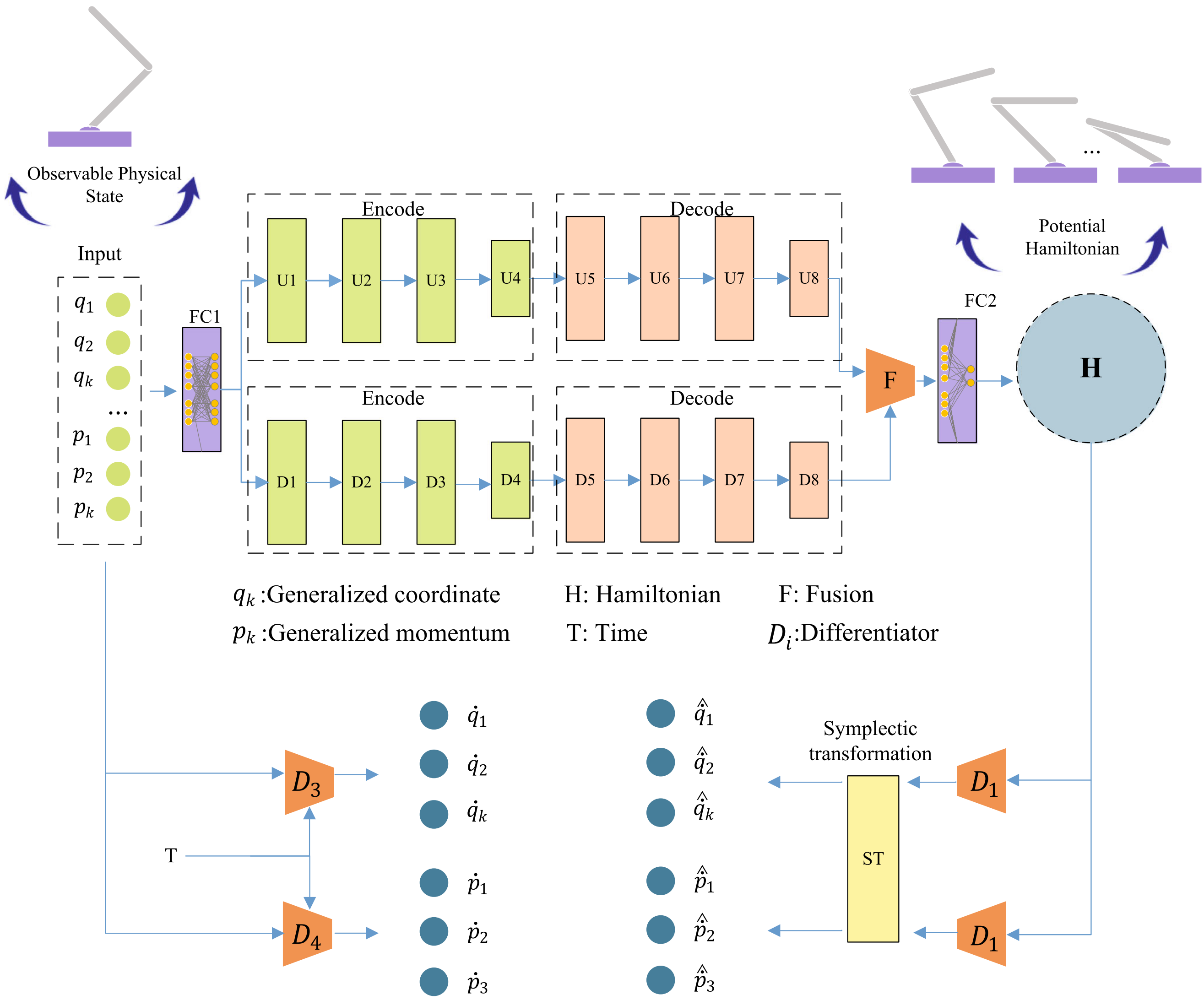


Fig. 2 Model structure diagram

where $\nabla_z H(z) = \partial H(z)/\partial z$. These numerical methods are called symplectic methods and have been widely used to calculate the long-term evolution of chaotic systems [37]. In the traditional neural network, the output is the predicted value of the derivative of the input w.r.t. time, but in the Hamilton neural network, the output is a Hamilton function that does not contain time, so the actual loss function calculation is shown in Fig. 3.

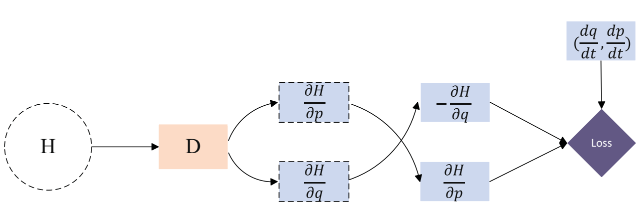


Fig. 3 Loss function calculation

The training loss of the model is the total loss calculated with the participation of Hamiltonian [38] as follows:

$$\mathcal{L}_{\mathcal{H}} = \frac{1}{K}\Sigma_{l=1}^{K}(\dot{\hat{z}}^{(n)} - S \cdot \nabla_{\hat{z}^{(n)}} H(\hat{z}^{(n)}))^2 \qquad (10)$$

The Hamiltonian with parametrization $\hat{z}^{(n)}$ is used in the loss function $Loss$, K is the number of training points and (n) indicates each time point.

The model parameter training uses a backward propagation algorithm, as shown below:

$$w^{(t+1)} \leftarrow w^{(t)} - \eta \frac{\partial Loss^{(t)}}{\partial w^{(t)}} \qquad (11)$$

where $w^{(t+1)}$ and $w^{(t)}$ are the connection weights of the $t+1$th and tth iterations, respectively. The minus sign indicates the direction of the gradient. η is the learning rate, generally $\eta \in (0, 1)$. The model can converge stably

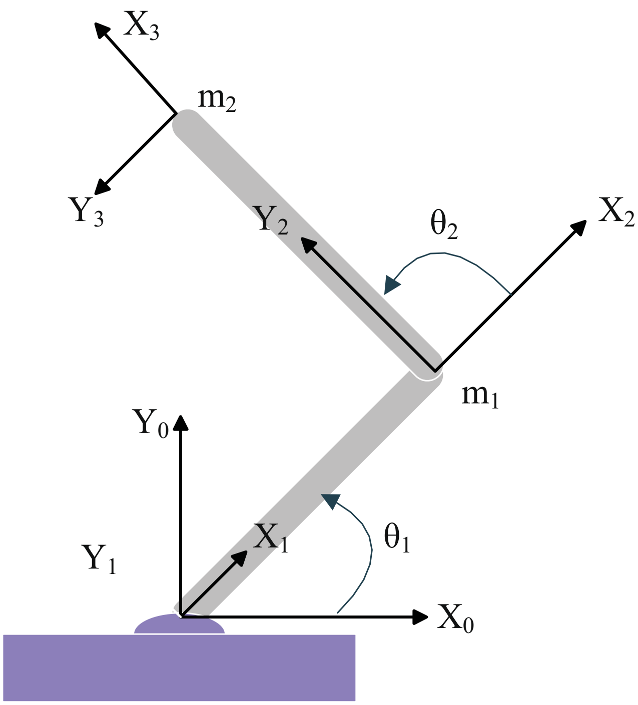


Fig. 4 Planar-linkage robot schematic

when η is relatively small, but the learning speed is slow, and a satisfactory solution is often not obtained due to the limitations of the training time. When η increases, the model learning speed will be accelerated, but the loss swing phenomenon may occur without convergence.

3 Experimental results

The experiment takes planar-linkage robot mechanisms as an example to comprehensively verify the effectiveness of the proposed model, as shown in Fig. 4. The relevant physical parameters of the robot system are shown in Table 1.

3.1 Classical dynamic analysis

To verify whether the neural network can actively learn the underlying laws of physics that the data obey, classical dynamics analysis is first performed. The kinetic energy and potential energy of joint 1 are as follows:

$$\begin{aligned} T_1 &= \frac{1}{2} m_1 (l_1 \dot{\theta}_1)^2 \\ V_1 &= m_1 g l_1 sin\theta_1 \end{aligned} \tag{12}$$

The position of mass m_2 is expressed as

$$\begin{aligned} x_2 &= l_1 \cos\theta_1 - l_2 \cos(\theta_1 + \theta_2) \\ y_2 &= l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) \end{aligned} \tag{13}$$

The velocity component will be obtained as follows:

$$\begin{aligned} \dot{x}_2 &= l_1 \sin\theta_1 \dot{\theta}_1 - l_2 \sin(\theta_1 + \theta_2)(\dot{\theta}_1 + \dot{\theta}_2) \\ \dot{y}_2 &= l_1 \cos\theta_1 \dot{\theta}_1 + l_2 \cos(\theta_1 + \theta_2)(\dot{\theta}_1 + \dot{\theta}_2) \end{aligned} \tag{14}$$

The square of the mass m_2 velocity is

$$\begin{aligned} v_2^2 &= \dot{x}_2^2 + \dot{y}_2^2 \\ &= (l_1 \sin\theta_1 \dot{\theta}_1 - l_2 \sin(\theta_1 + \theta_2)(\dot{\theta}_1 + \dot{\theta}_2))^2 \\ &\quad + (l_1 \cos\theta_1 \dot{\theta}_1 + l_2 \cos(\theta_1 + \theta_2)(\dot{\theta}_1 + \dot{\theta}_2))^2 \end{aligned} \tag{15}$$

In this way, the kinetic energy and potential energy of joint 2 are as follows:

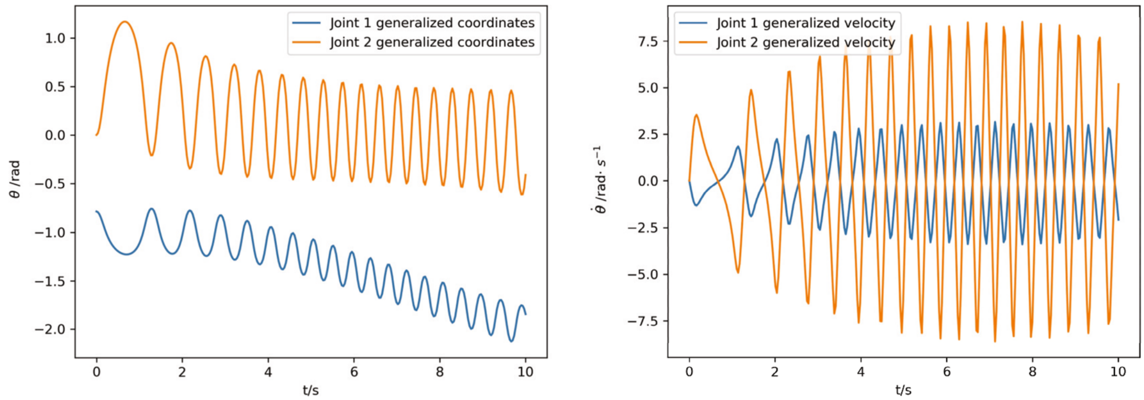
$$\begin{aligned} T_2 &= \frac{1}{2} m_2 [l_1^2 \dot{\theta}_1^2 + l_2^2 (\dot{\theta}_1^2 + 2\dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_2^2) + 2 l_1 l_2 \cos\theta_2 (\dot{\theta} 1_2 + \dot{\theta}_1 \dot{\theta}_2)] \\ V_2 &= m_2 g l_1 \sin\theta_1 + m_2 g l_2 \sin(\theta_1 + \theta_2) \end{aligned} \tag{16}$$

The Hamiltonian function is

$$\begin{aligned} L = & \frac{1}{2}(m_1 + m_2) l_1^2 \dot{\theta}_1^2 + \frac{1}{2} m_2 l_2^2 (\dot{\theta}_1 + \dot{\theta}_2)^2 \\ & + m_2 l_1 l_2 \cos\theta_2 (\dot{\theta}_1^2 + \dot{\theta}_1 \dot{\theta}_2) \\ & - (m_1 + m_2) g l_1 \sin\theta_1 - m_2 g l_2 \sin(\theta_1 + \theta_2) \end{aligned} \tag{17}$$

Table 1 Planar-linkage robot dynamic parameters

Parameter symbol	Parameter name	Parameter value	Physical quantities
m_1	Joint 1 mass	1	kg
m_2	Joint 2 mass	1	kg
l_1	Joint 1 length	1	m
l_2	Joint 2 length	1	m
θ_1	Joint 1 rotation angle	$(0, 2\pi)$	rad
θ_2	Joint 2 rotation angle	$(0, 2\pi)$	rad
g	Acceleration of gravity	9.8	ms^{-2}



(a) Generalized coordinate curve for q_1 and q_2

(b) Generalized velocity curve for dq_1 /dt and dq_2 /dt

Fig. 5 Generalized coordinate and generalized velocity simulation

To construct the Hamiltonian function H, first, the generalized momentum is solved by:

$$\begin{aligned} p_1 &= \frac{\partial L}{\partial \dot{\theta}_1} = (m_1 + m_2) l_1^2 \dot{\theta}_1 \\ &+ m_2 l_2^2 (\dot{\theta}_1 + \dot{\theta}_2) + m_2 l_1 l_2 \cos\theta_2 (2\dot{\theta}_1 + \dot{\theta}_2) \\ p_2 &= m_2 l_2^2 (\dot{\theta}_1 + \dot{\theta}_2) + m_2 l_1 l_2 \cos\theta_2 \dot{\theta}_1 \end{aligned} \tag{18}$$

Thus, the Hamiltonian function expression is shown:

$$\begin{aligned} H &= T_1 + V_1 + T_2 + V_2 \\ &= \frac{1}{2}(m_1 + m_2) l_1^2 \dot{\theta}_1^2 + \frac{1}{2} m_2 l_2^2 (\dot{\theta}_1 + \dot{\theta}_2)^2 \\ &+ m_2 l_1 l_2 \cos\theta_2 (\dot{\theta}_1^2 + \dot{\theta}_1 \dot{\theta}_2) + (m_1 + m_2) g l_1 \sin\theta_1 \\ &+ m_2 g l_2 \sin(\theta_1 + \theta_2) \end{aligned} \tag{19}$$

Then, $\dot{\theta}_1$ and $\dot{\theta}_2$ are replaced with generalized kinetic energy p_1 and p_2 by formula 16, forming a new function:

$$H = \phi(\theta_1, \theta_2, p_1, p_2) \tag{20}$$

where ϕ is the hidden function that the DPM will learn.

$$\mathcal{L} = \|\hat{\dot{q}} - \dot{q}\|_2 + \|\hat{\dot{p}} - \dot{p}\|_2 \tag{21}$$

where $(\hat{\dot{q}}, \hat{\dot{p}})$ is the actual output of the model, $(\dot{q}, \dot{p})$ is the derivative of the input w.r.t. time, $\dot{q} = \frac{\partial q}{\partial t}$, and $\dot{p} = \frac{\partial p}{\partial t}$. The Hamiltonian is calculated through first-order differential calculation, and the derivatives of the Hamiltonian w.r.t. q and p are obtained. Then, the matrix multiplication transformation is performed according to the derived potential Hamiltonian equation to obtain the corresponding derivatives of q, p w.r.t. time. Therefore, in Hamiltonian mechanics, a loss can be converted to the following formula combined with formula 6:

$$\mathcal{L}_c = \|\frac{\partial H}{\partial p} - \frac{\partial q}{\partial t}\|_2 + \|\frac{\partial H}{\partial q} + \frac{\partial p}{\partial t}\|_2 + \lambda \sum_l \|w\|_2^2 \tag{22}$$

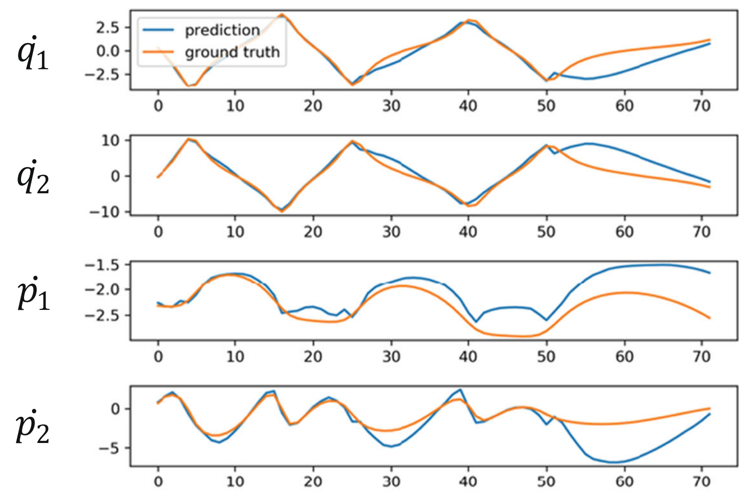
In the formula, $\frac{\partial H}{\partial q}$ and $\frac{\partial H}{\partial p}$ can be calculated by the first-order derivation of the output of the model w.r.t. the input (q, p). Through the simple mathematical calculation of the Hamiltonian independent of time, we can predict the derivative of the input physical quantity w.r.t. time. $L2$ regularization is added to reduce the overfitting problem.

Figure 5 shows the simulation results of the robot system in the initial state $(-\pi/4, 0, 0, 0)$ and gives a partial

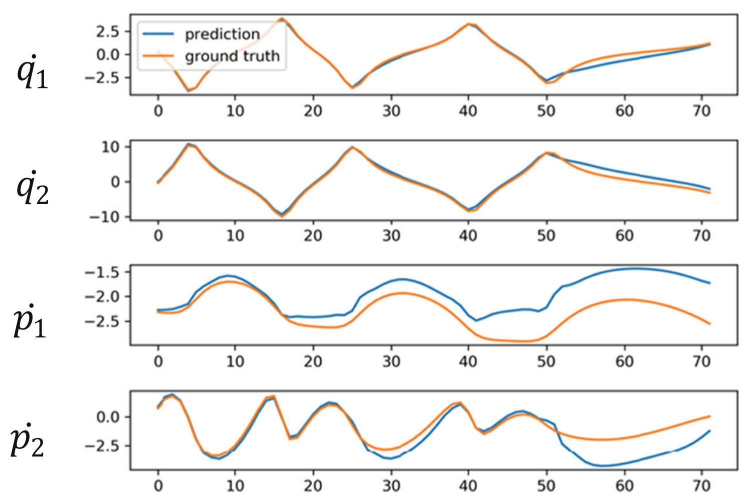
Table 2 Comparison of the training loss between the proposed model and the conventional neural network, DT represents the dynamic trajectory

Dynamic trajectory	$\mathcal{L}_{train}^{DPM}$	$\mathcal{L}_{test}^{DPM}$	$\mathcal{L}_{train}^{baseline}$	$\mathcal{L}_{test}^{baseline}$
DT1	0.31± 0.03	**9.6± 2.2**	1.02± 0.18	31±6.05
DT2	0.57± 0.08	**16± 3.1**	0.93± 0.14	26±5.03
DT3	0.30± 0.04	**11± 2.5**	2.74± 0.35	32±11.1
DT4	0.69± 0.10	46± 11	1.11± 0.17	**45±8.7**
DT5	0.47± 0.00	**9.9± 2.6**	4.05± 0.64	71±33
DT6	2.2± 0.6	**75± 21**	9.02± 2.59	465±92
DT7	0.46± 0.00	**39± 8.2**	0.47± 0.00	40±13
DT8	0.76± 0.00	**23± 5.7**	0.40± 0.00	85±30
DT9	0.19± 0.00	**86± 40**	0.02± 0.00	227±60
DT10	0.23± 0.00	**19± 4.2**	4.3± 0.55	87±27

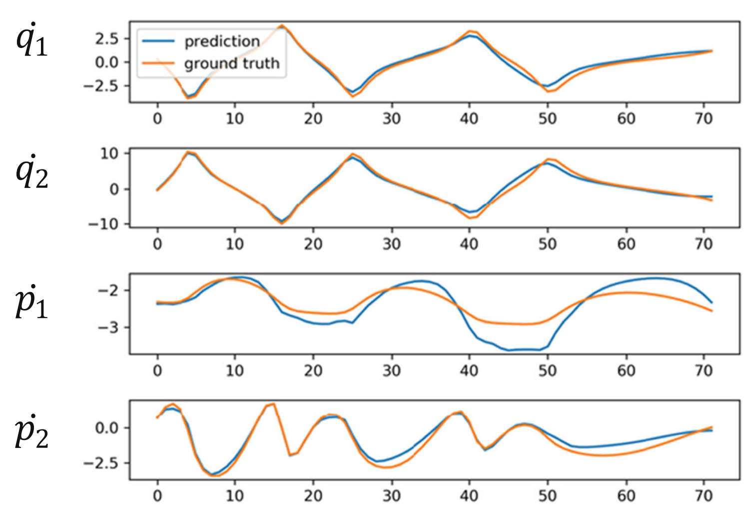
Fig. 6 Prediction performance comparison on system case 1



(a) Conventional neural network method



(b) HNN method



(c) DPM method

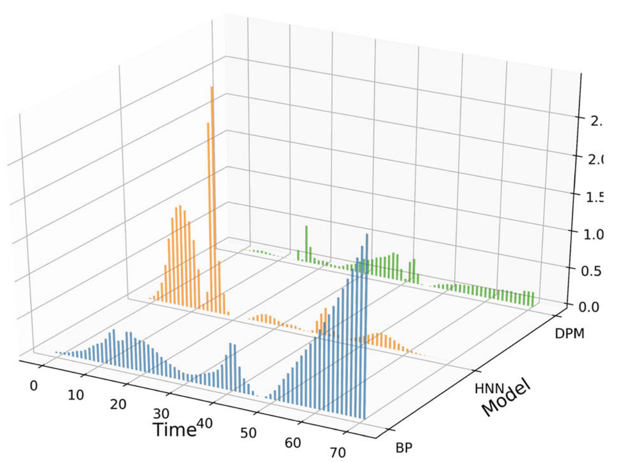
presentation of the kinetic data. Figure 5(a) represents the generalized coordinates as the input of the system potential energy, and Fig. 5(b) shows the generalized speed as the ground truth. The data are irregular and change over time.

Due to the constraints of structure and force, the joint movement generally changes in sinusoidal cycles. Different initial values will have different system responses, but their dynamic responses follow the rule of the Hamiltonian. The proportions of the training data and test data are 80% and 20%, respectively, and their input and output are different at any time. This model has not seen or learned any test data, but through learning the hidden Hamiltonian physical quantities, the model can still predict the dynamic response data. Next, the DPM is used to learn the abovementioned Hamiltonian rule and compare the results with SOTA methods, including the traditional neural network and the original HNN. Due to the uncertainty of the initial state of the robot, to obtain a more generalized result, a number of motion data satisfying the Hamiltonian equation can be obtained by setting different initial states. The observable q and p are sent to the DPM, and the output is a virtual Hamiltonian H_{ϕ}, no longer $\dot{q}$, $\dot{p}$. Through a series of differentiation and matrix transformations, the model converts the derivative of the Hamiltonian w.r.t. the input to the derivative of the input w.r.t. time, thereby continuously approximating the actual value and finally learning the parameters of the Hamiltonian.

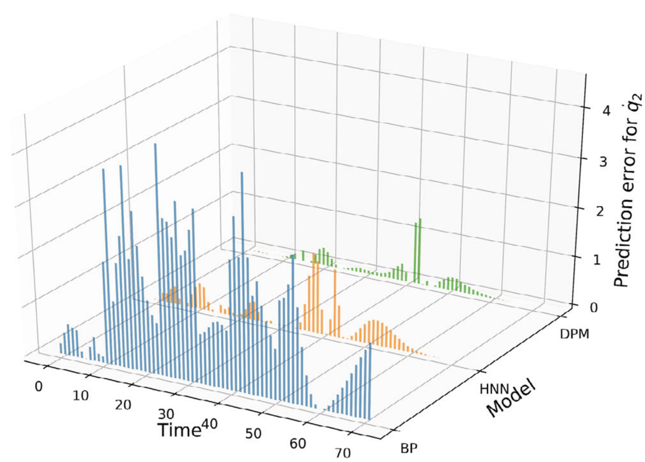
3.2 Comparative analysis of the DPM

3.2.1 Convergence performance

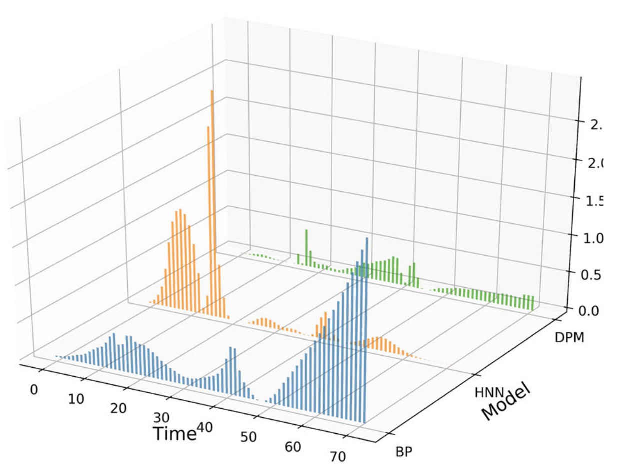
Table 2 shows the comparison of the mean squared error (MSE) during training and testing between the DPM and



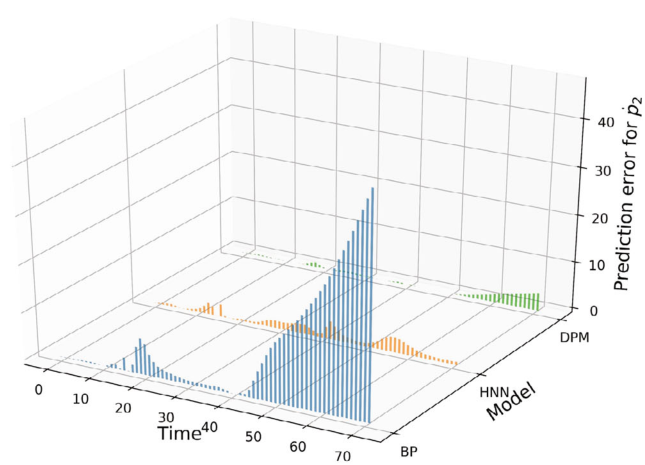
（a）Comparison of prediction errors ($\dot{q}_1$)



（b）Comparison of prediction errors ($\dot{q}_2$)



（c）Comparison of prediction errors ($\dot{p}_1$)



（d）Comparison of prediction errors ($\dot{p}_2$)

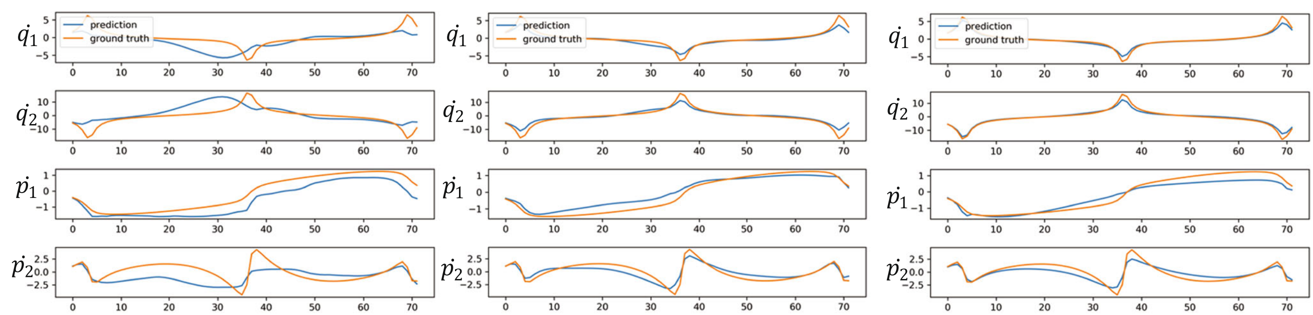
Fig. 7 Prediction error comparison on different models a conventional neural network. The variation of loss in network training reflects the performance of network fitting. The lower the loss is, the closer the predicted value of the network is to the ground truth. The baseline represents the conventional neural network, where a fully connected neural network is selected.

All loss values have been multiplied by 10^{-2} for better display. Each trajectory number represents the training situation in different initial states. The DPM is competitive with conventional neural networks in terms of training/test loss for the task of predicting chaotic data. Both training and testing losses of the DPM can be approximately an order of magnitude lower than those of the conventional neural network; in particular, the bold value in the table represents the lowest loss for unknown test data.

3.2.2 Performance comparison

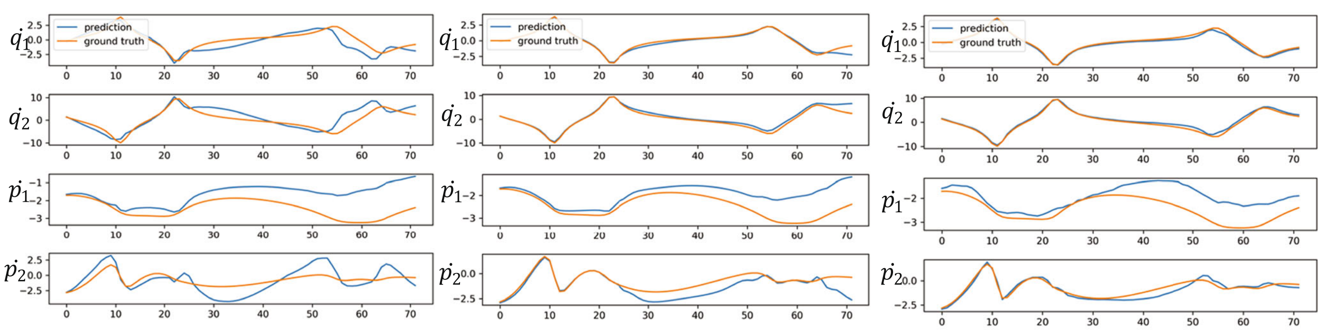
After completing enough iterative training, the model can learn some underlying physical laws that the data follow. Figure 6 shows a prediction performance comparison of the conventional neural network, HNN, and the DPM proposed

System performance Case 2



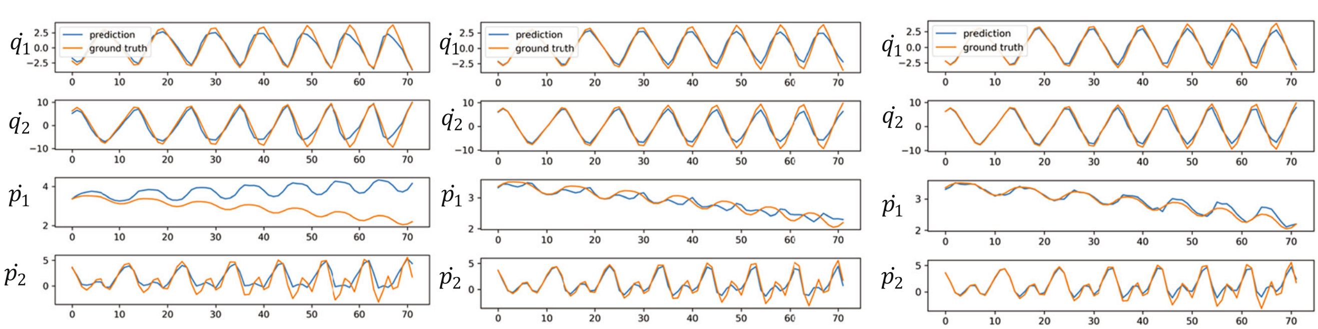
(a) Conventional neural network method (b) HNN method (c) DPM method

System performance Case 3



(a) Conventional neural network method (b) HNN method (c) DPM method

System performance Case 4



(a) Conventional neural network method (b) HNN method (c) DPM method

Fig. 8 Prediction performance comparison of different system cases by this paper. Figure 6(a) shows the prediction results of the conventional neural network model. Conventional neural networks perform well on general tasks; however, in chaotic system conditions, although with good performance in training, they have obvious shortcomings in predicting unknown data. The output of the conventional neural network is the four physical quantities $[\dot{q}_1, \dot{q}_2, \dot{p}_1, \dot{p}_2]$ that need to be predicted. The conventional neural network uses numerical calculation methods to find the optimal parameters. As seen in the figure, the predicted value deviates greatly from the ground truth because the underlying rules that the data in the chaotic system follow are not discovered. Figure 6(b) is a demonstration of the prediction results based on the existing Hamiltonian neural network method. Due to learning the underlying laws of physics, the prediction of chaotic complex systems performs better than that of conventional neural networks. The curve of $[\dot{q}_1, \dot{q}_2]$ is more consistent; however, the prediction of the time derivative of generalized momentum $[\dot{p}_1, \dot{p}_2]$ is not satisfactory. The reason for this is that the learning ability of the Hamiltonian quantity is not strong enough, so there will be calculation error when using the Hamiltonian rule. Figure 6(c) shows the excellent prediction result based on the proposed DPM, which indicates that the proposed model scaled well to this system. Although facing complex chaotic systems, in terms of generalized velocity prediction, $\dot{q}_1$ and $\dot{q}_2$ can completely fit the ground truth. Regarding the prediction of the momentum derivative w.r.t. time, the predicted data maintain the changes in momentum. The DPM multilayer stacking

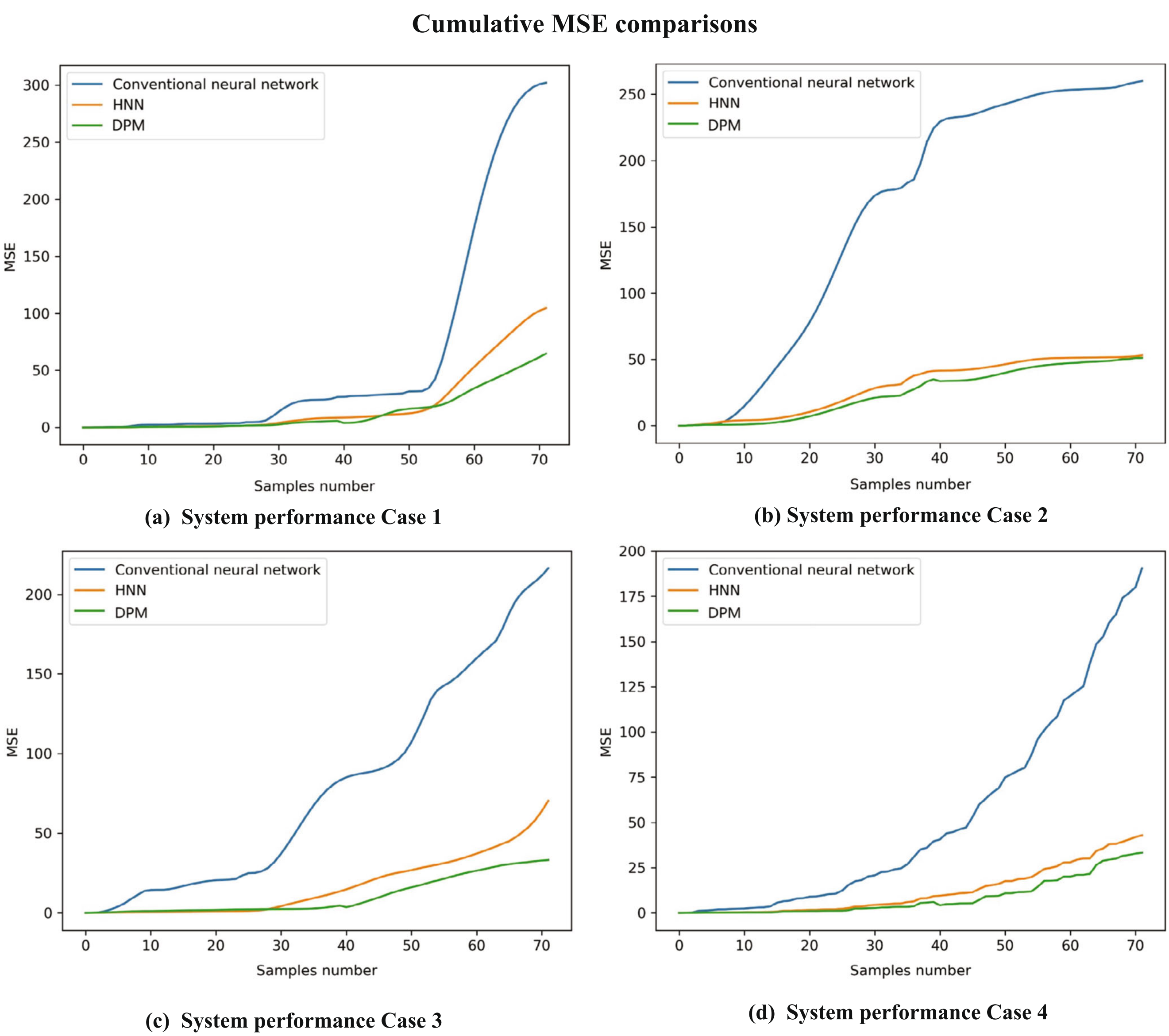


Fig. 9 Cumulative MSE comparisons

network enhances the learning ability of the Hamiltonian, making it perform better than other models in predicting $[\dot{q}_1, \dot{q}_2, \dot{p}_1, \dot{p}_2]$. Figure 7 represents the prediction error of the dynamic state $\dot{q}_1, \dot{q}_2, \dot{p}_1, \dot{p}_2$ under different models. For the task of predicting unknown chaotic data, the DPM proposed in this article has the highest accuracy. Figure 8 shows the comparison of prediction performance under different system conditions. Different cases correspond to different initial values. In terms of the overall prediction error, the prediction performance based on the DPM is better, and the cumulative MSE can be several orders of magnitude lower than that of the conventional neural network, as shown in Fig. 9. With its capability of learning Hamiltonian regular equations, it has an excellent predictive ability for chaotic data.

3.2.3 Discussion

The DPM works best for the prediction of chaotic dynamic systems. There are two reasons for this. On the one hand, instead of directly predicting dynamic physical quantities by means of numerical simulation through a neural network, the DPM is used to learn potential Hamiltonian functions to understand the Hamiltonian law of the system and achieve better prediction performance under chaotic data. On the other hand, the DPM stacks different and independent autoencoding neural networks and merges them to form multilayer embedding features. It learns the physical distribution rule of a certain latent variable, which is the Hamiltonian function, better than the existing HNN. The performance of chaotic dynamics data prediction is better than that of existing HNNs. In terms of the system prediction error, as time increases, the DPM has the smallest cumulative MSE. This is due to its stronger learning ability of the Hamiltonian, which is more effective for predicting time-independent parameters for time-dependent physical quantities.

4 Conclusion

Aiming at the limitations of conventional neural network prediction ability in the state of complex and chaotic robot dynamics data, a data-driven physical law learning model called the DPM is proposed, which can learn the underlying physical rules that the data follow. First, two independent autoencoder neural networks are stacked and merged to explore the potential physical rules. Then, a virtual Hamiltonian is introduced to represent the sum of the kinetic energy and the potential energy of the chaotic data. Combined with the Hamiltonian equation, the learned Hamiltonian is transformed into a symplectic transformation, whose first-order differential w.r.t. the generalized coordinates and momentum, can be regarded as the time-dependent prediction instead of direct numerical approximation. Finally, the DPM continuously learns implicit Hamiltonian equations from chaotic data until it can fit the law of phase space motion in a chaotic environment. The experimental results show that the model has better robot dynamics prediction ability in long-term chaotic systems than the existing SOTA methods.

Acknowledgments This paper is supported by the Nanjing Institute of Technology High-level Scientific Research Foundation for the introduction of talent (No. YKJ201918), the Natural Science Foundation Youth Fund of Jiangsu Province of China(No.BK20210931). The Natural Science Fondation of the Jiangsu Higher Education Institutions of China (No. 20KJB510049), and is partially supported by the National Natural Science Foundation of China (No. 61902179).

References

1. Featherstone R, Orin DE (2016) Dynamics[M]. Springer Handbook of Robotics. Springer, Cham, pp 37–66
2. Godois LM, Adamatti DF, Emmendorfer LR (2020) A multi-agent-based algorithm for data clustering. Prog Artif Intell 9(4):305–313
3. Wei C, Hindriks KV, Jonker CM (2016) Dynamic task allocation for multi-robot search and retrieval tasks. Appl Intell 45(2):383–401
4. Sousa CD, Cortesao R (2019) Inertia tensor properties in robot dynamics identification: A linear matrix inequality approach[J]. IEEE Trans Mechatron 24(1):406–411
5. Saadatzi M, Long DC, Celik O (2019) Comparison of human-robot interaction torque estimation methods in a wrist rehabilitation exoskeleton[J]. J Intell Robot Syst 94(3-4):565–581
6. Miclosina CO, Cojocaru V, Korka ZI (2015) Dynamic simulation of a parallel topology robot operation[J]. Appl Mech Mater 762:107–112
7. Xu D, Wu X, Chen YL et al (2015) Online dynamic gesture recognition for human robot interaction[J]. J Intell Robot Syst 77(3-4):583–596
8. Pan H, Dai J, Chen L et al (2014) Multi-robot parallel dynamic bounding volume hierarchy tree collision detection algorithm[J]. J Computer-Aided Des Comput Graphics 26(11):1948–1956
9. Vanraj GD, Saini A et al (2016) Intelligent predictive maintenance of dynamic systems using condition monitoring and signal processing techniques - A review[C]. In: 2016 International conference on advances in computing, communication, & automation (ICACCA) (Spring), IEEE
10. Liu Z, Wang X, Cai Y et al (2020) Dynamic risk assessment and active response strategy for industrial human-robot collaboration[J]. Comput Ind Eng 141:106302
11. Sciavicco L, Siciliano B, Villani L (1995) Lagrange and newton-euler dynamic modeling of a gear-driven robot manipulator with inclusion of motor inertia effects[J]. Adv Robot 10(3):317–334
12. Lu S, Zhao J, Jiang L et al (2017) Solving the time-jerk optimal trajectory planning problem of a robot using augmented lagrange constrained particle swarm optimization[J]. Math Probl Eng 2017(pt.6):1–10
13. Morabito F, Teel AR, Zaccarian L (2004) Nonlinear antiwindup applied to Euler-Lagrange systems[J]. IEEE Trans Robot Autom 20(3):526–537

14. Rahmani B, Belkheiri M (2019) Adaptive neural network output feedback control for flexible multi-link robotic manipulators[J]. Int J Control 92(10):2324–2338
15. Razzaghi P, Al Khatib E, Hurmuzlu Y (2019) Nonlinear dynamics and control of an inertially actuated jumper robot[J]. Nonlinear Dyn 97(1):161–176
16. Lu Y, Yan D, Zhou M et al (2017) Maximum likelihood parameter estimation of dynamic systems by heuristic swarm search[J]. Intell Data Anal 21(1):97–116
17. Borovykh A, Oosterlee CW, Bohté SM (1020) Generalization in fully-connected neural networks for time series forecasting[J]. J Comput Sci 36(10):2019
18. Jiang Y, Chen J, Zhou H et al (2021) Residual learning of the dynamics model for feeding system modelling based on dynamic nonlinear correlate factor analysis. Appl Intell 51:5067–5080. https://doi.org/10.1007/s10489-020-02096-2
19. Wang Z, Goldsmith P, Gu J (2009) Adaptive trajectory tracking control for Euler-Lagrange systems with application to robot manipulators[J]. Control Intell Syst 37(1):46–56
20. Valverde A, Tsiotras P (2018) Modeling of spacecraft-mounted robot dynamics and control using dual quaternions[C]. In: IEEE Annual american control conference, 2018 pp 670–675
21. Liang B, Li T, Chen Z et al (2018) Robot arm dynamics control based on deep learning and physical simulation[C]. In: 2018 37th Chinese control conference (CCC). IEEE, pp 2921–2925
22. Bae HJ, Jin M, Suh J et al (2017) Control of robot manipulators using time-delay estimation and fuzzy logic systems[J]. J Electr Eng Technol 12(3):1271–1279
23. Sun F, Sun Z, Woo PY (2001) Neural network-based adaptive controller design of robotic manipulators with an observer[J]. IEEE Trans Neural Netw 12(1):54–67
24. Mbede JB, Wei W, Zhang Q (2001) Fuzzy and recurrent neural network motion control among dynamic obstacles for robot manipulators[J]. J Intell Robot Syst 30(2):155–177
25. Chu M, Song JZ, Jia QX et al (2013) Intelligent control for model-free robot joint with dynamic friction using wavelet neural networks. J Theor Appl Inf Technol 50(1):167–173
26. Saleki A, Fateh MM (2020) Model-free control of electrically driven robot manipulators using an extended state observer[J]. Comput Electr Eng 87:106768
27. Asl HJ, Janabi-Sharifi F (2017) Adaptive neural network control of cable-driven parallel robots with input saturation[J]. Eng Appl Artif Intell 65:252–260
28. Su H, Qi W, Yang C et al (2020) Deep neural network approach in robot tool dynamics identification for bilateral teleoperation[J]. IEEE Robot Autom Lett 5(2):2943–2949
29. Luan F, Na J, Huang Y et al (2019) Adaptive neural network control for robotic manipulators with guaranteed finite-time convergence[J]. Neurocomputing 337:153–164
30. Karpatne A, Atluri G, Faghmous JH et al (2017) Theory-guided data science: A new paradigm for scientific discovery from data[J]. IEEE Trans Knowl Data Eng 29(10):2318–2331
31. Zhang P, Shen H, Zhai H (2018) Machine learning topological invariants with neural networks[J]. Phys Rev Lett 120(6): 066401
32. Moret-Bonillo V (2018) Emerging technologies in artificial intelligence: quantum rule-based systems[J]. Prog Artif Intell 7(2):155–166
33. Doan N, Polifke W, Magri L (2020) Physics-informed echo state networks[J]. J Comput Sci 47:101237
34. Dominguez DRC, Korutcheva E (2000) Three-state neural network: From mutual information to the Hamiltonian[J]. Phys Rev 62(2 Pt B):2620–2628
35. Yang B, Li HG, Sha XP et al (2012) A speed observer for robot based on hamiltonian theory and immersion & invariance[J]. Acta Automatica Sinica 38(11):1757
36. Rabinowitz PH (2010) Periodic solutions of Hamiltonian systems and related topics[J]. Commun Pure Appl Math 31(2):157–184
37. Leimkuhler BJ, Skeel RD (1994) Symplectic numerical integrators in constrained hamiltonian systems[J]. J Comput Phys 112(1):117–125
38. Greydanus SJ, Dzamba M, Yosinski J (2019) Hamiltonian neural networks. In: 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada
39. Bertalan T, Dietrich F, Mezic I et al (2019) On learning Hamiltonian systems from data[J]. Chaos: An Interdiscip J Nonlinear Sci 29(12):121107
40. Miller ST, Lindner JF, Choudhary A, Sinha S, Ditto WL (2020) The scaling of physics-informed machine learning with data and dimensions. Chaos, Solitons & Fractals: X 5:100046
41. Shirvany Y, Hayati M, Moradian R (2009) Multilayer perceptron neural networks with novel unsupervised training method for numerical solution of the partial differential equations[J]. Appl Soft Comput 9(1):20–29

Kui Qian received the Ph.D. in Instrument Science and Technology from Southeast University, Nanjing, China in 2014. He is currently an associate professor in Nanjing Institute of Technology. His research interests include mobile robot, deep learning and computer vision.

Lei Tian received the Ph.D. in Instrument Science and Technology from Southeast University and the M.S. in Measurement Technology and Instruments from Jilin University, China. She is currently an associate professor in Nanjing Institute of Technology. Her research interests include virtual reality, and human-computer interaction.