Physics of Fluids

A novel spatial-temporal prediction method for unsteady wake flows based on hybrid deep neural network

Cite as: Phys. Fluids **31**, 127101 (2019); https://doi.org/10.1063/1.5127247
Submitted: 10 September 2019 • Accepted: 06 November 2019 • Published Online: 03 December 2019

Renkun Han (韩仁坤), **Yixing Wang** (王怡星), **Yang Zhang** (张扬), **et al.**

COLLECTIONS

This paper was selected as an Editor's Pick

View Online Export Citation CrossMark

ARTICLES YOU MAY BE INTERESTED IN

Fast flow field prediction over airfoils using deep learning approach
Physics of Fluids **31**, 057103 (2019); https://doi.org/10.1063/1.5094943

Deep learning methods for super-resolution reconstruction of turbulent flows
Physics of Fluids **32**, 025105 (2020); https://doi.org/10.1063/1.5140772

Machine learning methods for turbulence modeling in subsonic flows around airfoils
Physics of Fluids **31**, 015105 (2019); https://doi.org/10.1063/1.5061693

AIP Publishing

© 2019 Author(s).

Physics of Fluids ARTICLE scitation.org/journal/phf

A novel spatial-temporal prediction method for unsteady wake flows based on hybrid deep neural network

Cite as: Phys. Fluids 31, 127101 (2019); doi: 10.1063/1.5127247
Submitted: 10 September 2019 • Accepted: 6 November 2019 •
Published Online: 3 December 2019

View Online Export Citation CrossMark

Renkun Han (韩仁坤),[1,2] Yixing Wang (王怡星),[1,2,3] Yang Zhang (张扬),[1] and Gang Chen (陈刚)[1,2,a)]

AFFILIATIONS

[1]State Key Laboratory for Strength and Vibration of Mechanical Structures, School of Aerospace Engineering, Xi'an Jiaotong University, Xi'an 710049, China
[2]Shannxi Key Laboratory for Environment and Control of Flight Vehicle, Xi'an Jiaotong University, Xi'an 710049, China
[3]The Key Laboratory of Reliability and Environment Engineering, Beijing 100076, China

[a)]Author to whom correspondence should be addressed: aachengang@mail.xjtu.edu.cn

ABSTRACT

A fast and accurate prediction method of unsteady flow is a challenge in fluid dynamics due to the high-dimensional and nonlinear dynamic behavior. A novel hybrid deep neural network (DNN) architecture was designed to capture the spatial-temporal features of unsteady flows directly from high-dimensional numerical unsteady flow field data. The hybrid DNN is constituted by the convolutional neural network, convolutional long short term memory neural network, and deconvolutional neural network. The unsteady wake flow around a cylinder at various Reynolds numbers and an airfoil at a higher Reynolds number are calculated to establish the datasets as training samples of the hybrid DNN. The trained hybrid DNNs were then tested by predicting the unsteady flow fields in future time steps. The predicted flow fields using the trained hybrid DNN are in good agreement with those calculated directly by a computational fluid dynamic solver.

Published under license by AIP Publishing. https://doi.org/10.1063/1.5127247

I. INTRODUCTION

High fidelity modeling of unsteady flows is one of the major challenges in computational physics. The high fidelity computational fluid dynamics (CFD) techniques have made significant inroads into this problem. However, such high fidelity numerical simulations with billions of degrees of freedom in order to resolve the energetically relevant spatial and temporal scales lead to time-consuming computations. Sometimes the high-dimensional and large data obtained from the time-consuming CFD simulations are difficult to directly use in optimization and flow control applications. Therefore, data-driven low-dimensional models that can capture the main dynamic characteristics of unsteady dynamic systems with good efficiency and enough accuracy become a very interesting concern.[1]

Reduced order modeling (ROM), as one of the model identification methods, has proven to be a powerful tool to reduce the complexity and large dimensional size of the dynamical system.[2] ROM, such as proper orthogonal decomposition (POD)[3] and dynamic mode decomposition (DMD),[4] offers the potential to simulate physical and dynamic systems with substantially increased computational efficiency while maintaining reasonable accuracy. A lot of research[5–8] has been carried out to analyze flow fields with low-dimensional representations using these methods. However, most of these investigations are linear or weakly nonlinear methods with some strong assumptions, which limit the applications of these methods to more complex unsteady flows.

Deep learning technology[9] is a recent advancement in artificial neural networks which is capable of finding more complex and hidden information from the big data. It has the advantage of learning the nonlinear system with multiple levels of representation data. Recently, it has resulted in breakthroughs in various areas such as image processing,[10] speech recognition,[11] and disease diagnosis.[12] More recently, there is a great interest in introducing the deep learning method to fluid mechanics. The pioneering investigation of the deep learning technology for turbulence modeling with Reynolds-averaged Navier-Stokes (RANS) simulations has been conducted.[13–16] These methods increased the accuracy of

Published under license by AIP Publishing

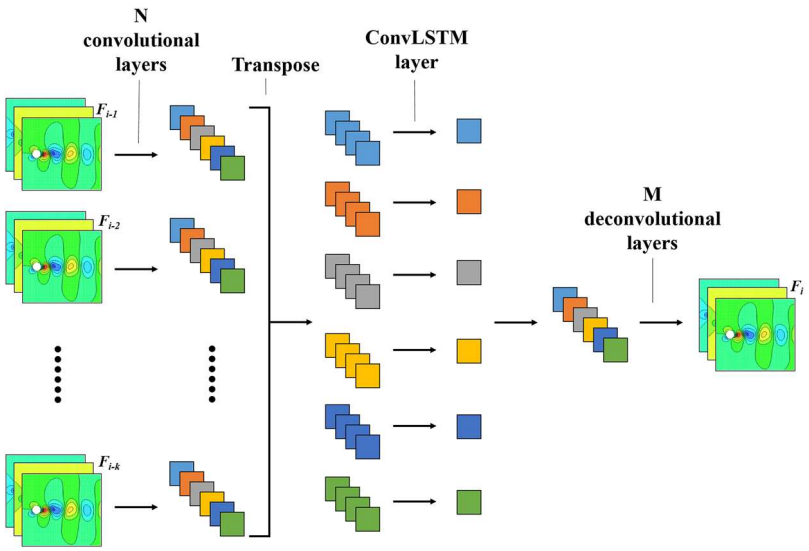

FIG. 1. The architecture of the hybrid deep neural network.

RANS models by utilizing neural networks to learn Reynolds stress closures.

However, there is still an urgent need to model the spatial-temporal dynamics of unsteady flows at a low computational cost. Compared with traditional ROMs, the deep learning based ROM can capture more nonlinear features of mapping data without loss of information while incorporating the appropriate degree of nonlinearity. In the work of Wang *et al.*[17] and Omata *et al.*,[18] the deep convolutional autoencoder data-driven nonlinear low-dimensional representation method was used for a dimensionality reduction in unsteady flow fields. Fukami *et al.*[19] reconstructed the high-resolution high-dimensional flow fields from under-resolved low-dimensional turbulent flow field data by the CNN and the hybrid downsampled skip-connection multiscale models. Jin *et al.*[20] predicted the velocity field around a cylinder using the pressure field on the cylinder by fusion convolutional neural networks (CNNs). Sekar *et al.*[21] use the deep Convolutional Neural Network (CNN) and deep Multilayer Perceptron (MLP) to predict incompressible laminar steady flow field over airfoils. However, in these works, the deep learning networks were only used to do dimensionality reduction or reconstruct the current time flow fields as the training flow data, which is not able to model the spatial-temporal flow dynamics simultaneously and predict unsteady advancing flow fields at future occasions.

To capture the temporal features of unsteady flow by deep neural networks (DNNs), Mohan and Gaitonde,[22] Pawar *et al.*,[23] and Deng *et al.*[24] constructed ROMs by combining the POD method and the Long Short Term Memory (LSTM) network. The POD method was used to reduce the dimensional size of flow fields and represent flow fields by the combination of POD models. The LSTM network was used to predict the time coefficients of the flow POD modes. Another combination of the CNN and convolutional Long Short Term Memory (ConvLSTM) is also been proposed by Mohan *et al.* for dimensionality reduction and spatial-temporal modeling of the dynamics of turbulence.[25] Not only LSTM, the generative adversarial networks (GANs)[26,27] are also been used to predict unsteady flow fields at future occasions.

In this paper, a novel hybrid deep neural network is designed to achieve fast and accurate prediction of spatial-temporal features of unsteady flow fields simultaneously. We will take advantage of the latest research progress in the reverse of CNN, ConvLSTM, and deconvolutional neural network (DeCNN) with the goal of dimensionality reduction in high dimensional unsteady flow datasets and try to model the spatial-temporal dynamics characteristics at a low computational cost. Different from previous work, we try to use only one integrated deep neural network to reduce dimensionality of unsteady flow fields and capture the spatial-temporal flow dynamic behavior simultaneously. More specifically, a novel hybrid deep neural network is designed to capture the complex spatial-temporal flow dynamics mapping directly from the high-dimensional flow fields without using any explicit intermediate dimensionality reduction method and can predict the unsteady flow fields at future occasions based on the captured features from the flow data at past times.

The structure of this article is as follows: Sec. II introduces the architecture of the proposed hybrid deep neural network

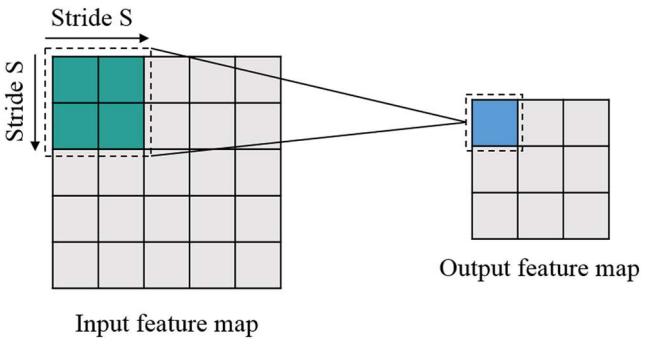

FIG. 2. Convolutional layer.

Published under license by AIP Publishing

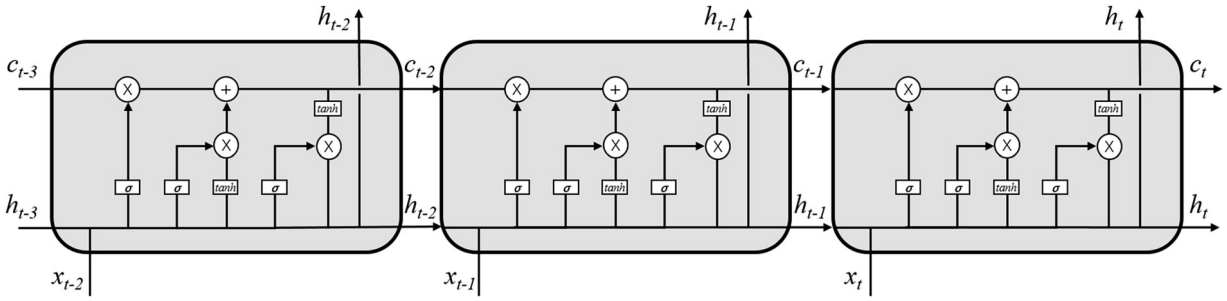

FIG. 3. LSTM layout with cell connections.

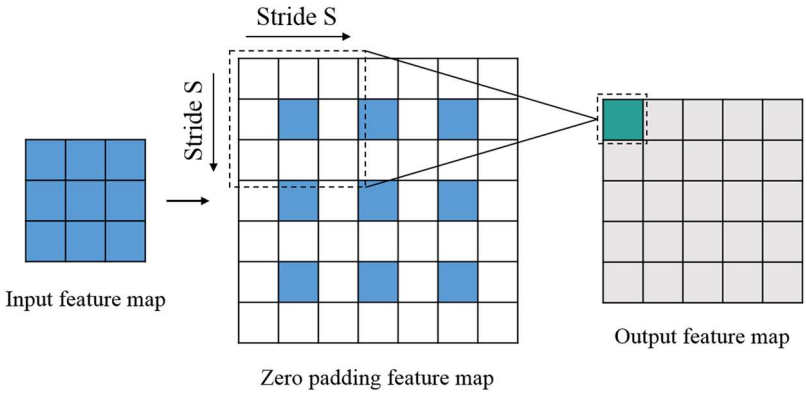

FIG. 4. Deconvolutional layer.

architecture. Then, in Sec. III, the method for constructing flow field dataset and the neural network training algorithm are explained. Section IV evaluates the performance of the proposed hybrid deep neural network. Finally, a summary and conclusion is provided in Sec. V.

II. THE ARCHITECTURE OF THE HYBRID DEEP NEURAL NETWORK

A. Architectural design of the hybrid deep neural network

As explained in Sec. I, the goal of this work is to reduce the dimension of high dimensional unsteady flow data and to learn its spatial-temporal dynamic characteristics directly from the past time flow data with the deep neural network. The flow fields depend not only on the current state of motion but also on the time history of motion. Predicting flow field at future occasions is based on the time history of motion as shown in Eq. (1), in which F_i is the flow fields at time t_i. Therefore, a hybrid deep neural network architecture composed of CNN layers, the ConvLSTM layer, and DeCNN layers is designed to capture the spatial-temporal features of unsteady flows, as shown in Fig. 1,

$$F_i = f(F_{i-1}, F_{i-2}, \ldots, F_{i-k}). \tag{1}$$

The CNN layers, consisting of six convolutional layers, are designed to capture complex spatial features directly from the high-dimensional input fields and represent it in a low-dimensional form. Several features of each time step flow field are obtained by CNN layers. After CNN layers, a "reshape layer" is settled to reshape the shape of CNN layers' output matrix into the shape ConvLSTM layer's input should be, which means turning the

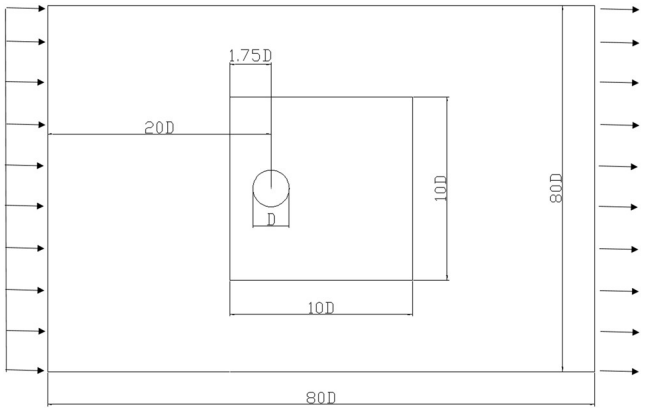

FIG. 5. The schematic diagram of the computational domain for numerical simulations (whole domain) and the training domain for collecting datasets (area inside the small square). *D* is the diameter of the circular cylinder or the chord length of the airfoil.

Published under license by AIP Publishing

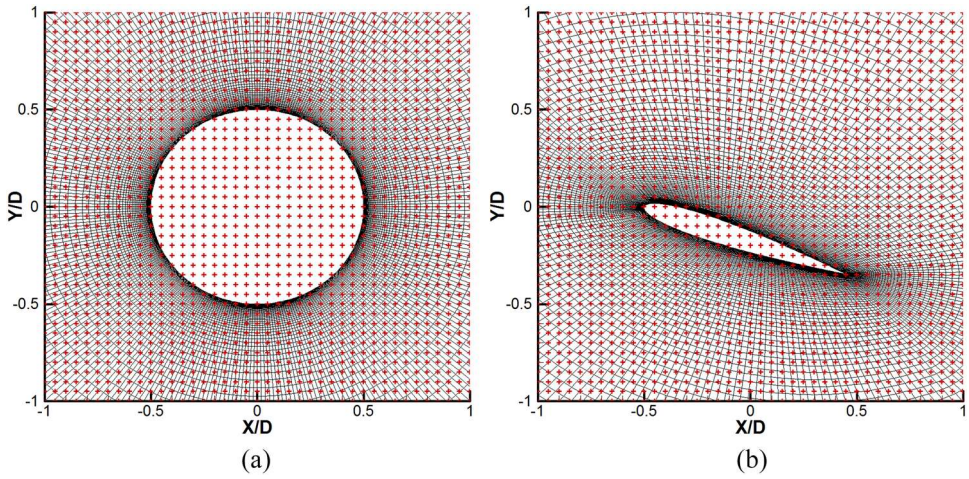

FIG. 6. The discrete structured mesh representation of a cylinder and an airfoil (in black) on a Cartesian grid (in red). (a) Grid for flow around a cylinder. (b) Grid for flow around an airfoil.

TABLE I. Details of the structure parameters in the hybrid deep neural network.

Layer	Kernel size/stride	Output size
Conv1	2 × 2/1	16 × 200 × 200 × 4
Conv2	3 × 3/2	16 × 100 × 100 × 8
Conv3	3 × 3/2	16 × 50 × 50 × 16
Conv4	3 × 3/2	16 × 25 × 25 × 32
Conv5	3 × 3/2	16 × 13 × 13 × 64
Conv6	3 × 3/2	16 × 7 × 7 × 64
Reshape1	...	64 × 7 × 7 × 16
ConvLSTM	3 × 3/1	64 × 7 × 7 × 1
Reshape2	...	1 × 7 × 7 × 64
DeConv6	3 × 3/2	1 × 13 × 13 × 64
DeConv5	3 × 3/2	1 × 25 × 25 × 32
DeConv4	3 × 3/2	1 × 50 × 50×16
DeConv3	3×3/2	1 × 100 × 100 × 8
DeConv2	3 × 3/2	1 × 200 × 200 × 4
DeConv1	2 × 2/1	1 × 200 × 200 × 3

"batch size" of CNN layers into the "time steps" of the "ConvLSTM layer." Then, a "ConvLSTM layer," eating convolved flow features, is designed to capture temporal features between low-dimensional features and predict the feature maps of flow fields at future occasions. For each kind of flow field feature map, the ConvLSTM layer predicts flow field feature map at future occasions based on corresponding feature maps of flow fields at previous occasions. The DeCNN layers copy the architecture of the CNN layers and reverse it. The DeCNN layers are used to represent the predicted low-dimensional feature maps to high-dimensional output field, with the same dimension as input fields. All those layers used to build the hybrid deep neural network are described in detail as follows.

B. Convolutional layer

The convolution operation extracts feature from images, enhancing certain features of the original signal and reducing noise.[28] The amount of data processing can be reduced while preserving useful information. A convolutional layer usually contains a plurality of feature maps with different weight vectors so that

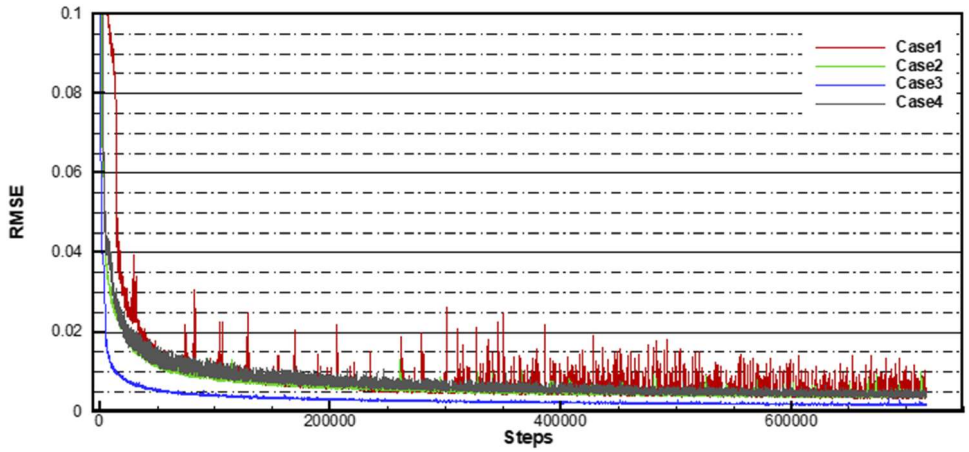

FIG. 7. Training error for four datasets decreases with increasing number of training steps.

Published under license by AIP Publishing

different features in the image can be preserved. The convolutional layer's parameters consist of a set of learnable filters (or kernels), which have a small receptive field, but extend through the full depth of the input volume. During the forward pass, each filter is convolved across the width and height of the input volume, computing the dot product between the entries of the filter and the input and producing a 2-dimensional activation map of that filter, as shown in Fig. 2. As a result, the network learns filters that activate when it detects some specific type of feature at some spatial position in the input. Stacking the activation maps for all filters along the depth dimension forms the full output volume of the convolution layer.

The convolutional layer and the nonlinear activation function compute the output feature map of the previous layer via a convolutional filtering operator,

$$y_c = \sigma(k * x_c + b_c), \tag{2}$$

where x_c is the input feature map of the convolutional layer, $*$ is the convolutional operator, k is the learnable convolutional kernel, b_c is the additive bias, σ is the nonlinear activation function, and y_c is the output feature map.

In this study, the leaky rectified linear unit (PReLU)[29] is used as the nonlinear activation function σ, i.e., a is a

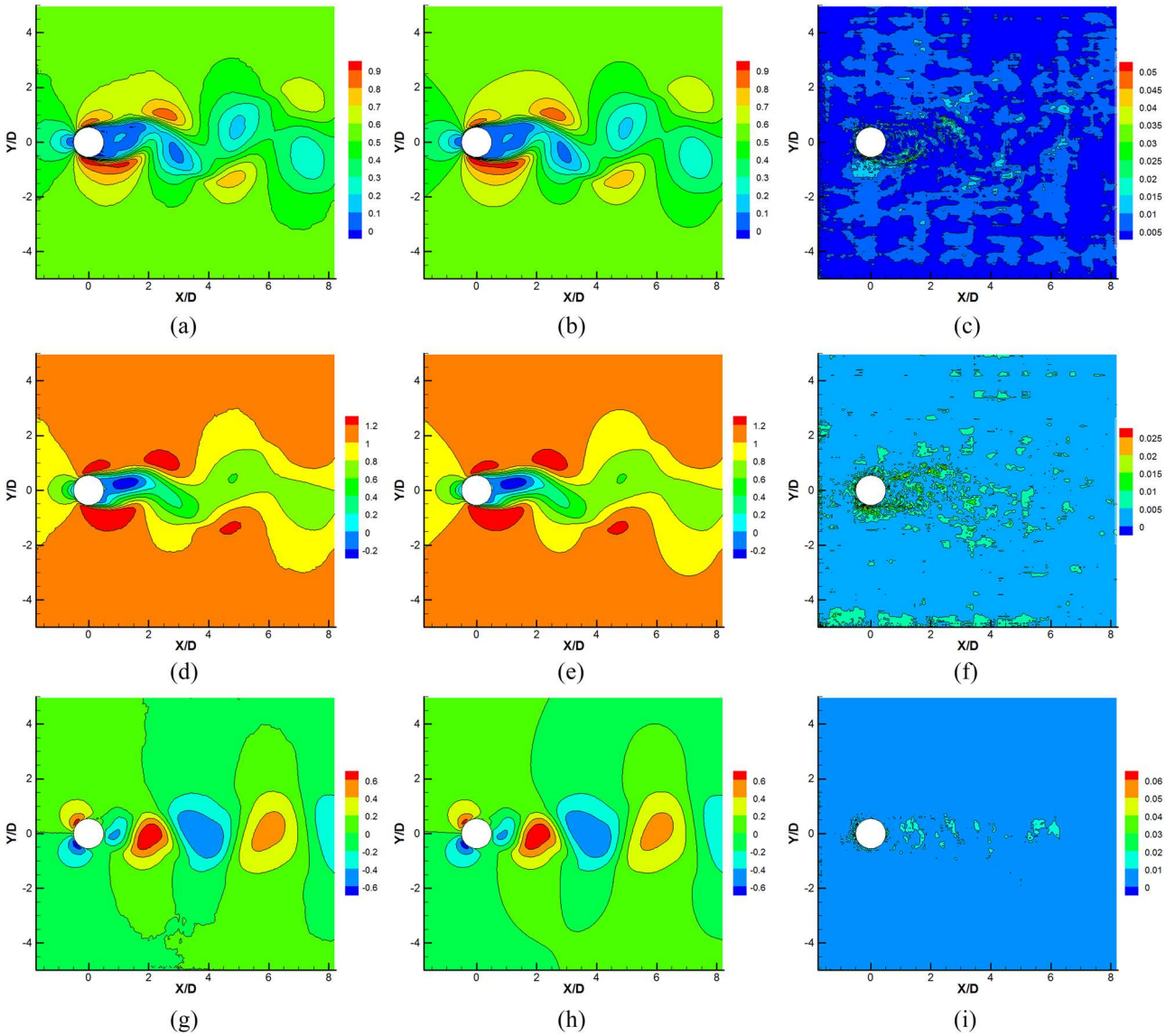

FIG. 8. Comparisons of instantaneous flow fields after 64 time-steps between the model predictions and CFD results for Case 1 (flow past a cylinder at Re = 200): network predictions for (a) pressure, (d) streamwise velocity, and (g) vertical velocity; CFD results for (b) pressure, (e) streamwise velocity, and (h) vertical velocity; and absolute prediction error for (c) pressure, (f) streamwise velocity, and (i) vertical velocity.

Published under license by AIP Publishing

constant with a very small value, and it is set to 0.02 in this paper,

$$\sigma(x) = PReLU(x) = \max(ax, x). \tag{3}$$

C. Convolutional long short term memory layer

The LSTM networks are specialized at capturing the temporal characteristics.[30] The typical LSTM cell contains three gates: the input gate, the output gate, and the forget gate. The LSTM cell regulates the flow of training information through these gates by selectively adding information (input gate), removing information (forget gate), or letting it through to the next cell (output gate). The LSTM networks, as shown in Fig. 3, utilize their internal memory such that the predictions are conditional on the recent context in the input sequence, not what has just been presented as the current input to the network. For instance, to predict the realization at time t_i, the LSTM networks can learn from the data at t_{i-1} and also at t_{i-k} since the outcome of the system depends on its previous realizations.

In the traditional LSTM cell, the input and hidden states consist of a one-dimensional vector; therefore, a two-dimensional input (such as an image or a data field) has to be resized to a single

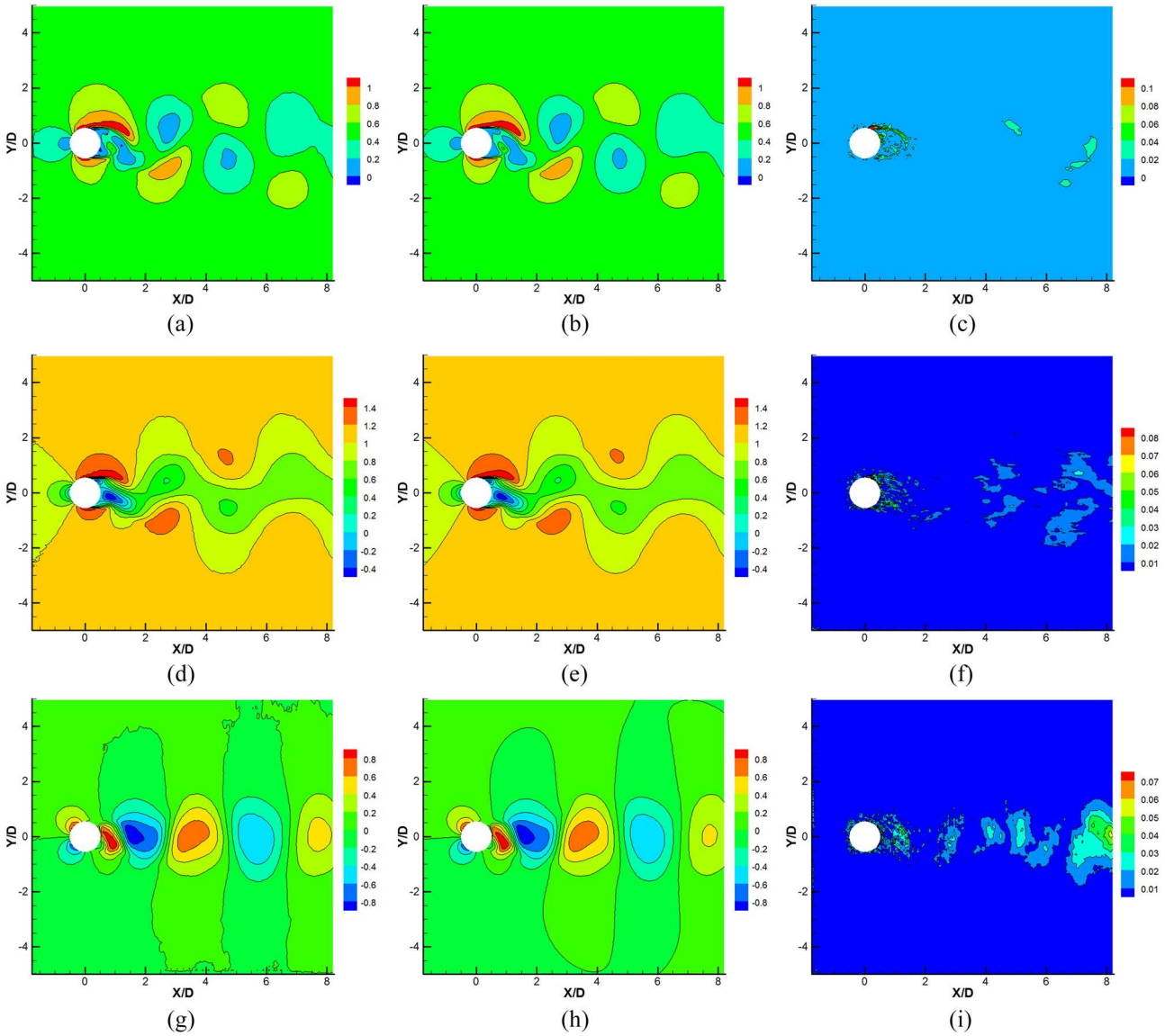

FIG. 9. Comparisons of instantaneous flow fields after 64 time-steps between the model predictions and CFD results for Case 2 (flow past a cylinder at Re = 4000): network predictions for (a) pressure, (d) streamwise velocity, and (g) vertical velocity; CFD results for (b) pressure, (e) streamwise velocity, and (h) vertical velocity; and absolute prediction error for (c) pressure, (f) streamwise velocity, and (i) vertical velocity.

Published under license by AIP Publishing

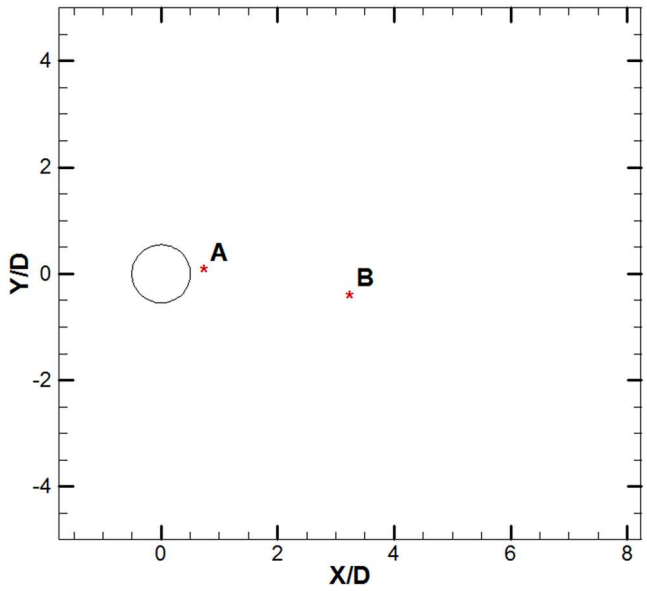

FIG. 10. A map of the positions selected in the wake to show the model prediction accuracy. The spatial coordinates of these points are described by dimensionless x* and y*: circle center (0, 0); point in the separating free layer A (0.75, 0) and point in the wake B (3.25, −0.5).

dimension. The "removal" of this dimensionality information fails to capture spatial correlations that may exist in such data, leading to increased prediction errors. In contrast, the ConvLSTM cell, proposed by Shi *et al.*,[31] could process hidden and input states in two dimensions, thereby retaining spatial information in the data. The ConvLSTM cell consists of a simple but powerful idea that the gates have the same dimensionality of the input data. This enables us to provide a 2D image input and obtain a 2D vector cell state as outputs from the ConvLSTM cell.

The input gate is represented by i, the output gate is represented by o, and the forget gate is represented by f. The cell state is represented as c, and the cell output is given by h, while the cell input is denoted as x. The weights for each of the gates are represented as W. Consider the equations of a ConvLSTM cell to compute its gates and states as shown in the following equation:

$$\begin{aligned} f_t &= \sigma\left(W_{xf} * \chi_t + W_{hf} * h_{t-1} + W_{cf} \odot c_{t-1} + b_f\right), \\ i_t &= \sigma\left(W_{xi} * \chi_t + W_{hi} * h_{t-1} + W_{ci} \odot c_{t-1} + b_i\right), \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tanh\left(W_{xc} * \chi_t + W_{hc} * h_{t-1} + b_c\right), \\ o_t &= \sigma\left(W_{xo} * \chi_t + W_{ho} * h_{t-1} + W_{co} \odot c_{t-1} + b_o\right), \\ h_t &= o_t \odot \tanh\left(c_t\right). \end{aligned} \tag{4}$$

D. Deconvolutional layer

The deconvolution operator, proposed by Zeiler,[32,33] was used for the visualization of every network layer learning result. With the successful application of deconvolution in neural network visualization, it has been adopted by more and more work such as scene segmentation. In mathematics, deconvolution is an algorithm-based process used to reverse the effect of convolution on recorded data. The essence of deconvolution is convolution, but with an automatic zero padding is added before the convolution, as shown in Fig. 4.

Deconvolutional layers multiply each element of the input with a filter (kernel) and sum over the resulting feature map, effectively swapping the forward and backward passes of a regular convolutional layer. The effect of using deconvolutional layers is to decode low-dimensional abstract features to a larger dimensional representation. It should be noted that the deconvolution cannot restore the matrix before convolution and can only restore the size.

III. TRAINING METHOD OF HYBRID DEEP NEURAL NETWORKS

A. Dataset constructions

Since the CNN is developed from the field of computer vision, we consider the goal of predicting flow fields at future occasions based on the time history of flow fields in a similar fashion to approaches considered in deep learning for image-to-image

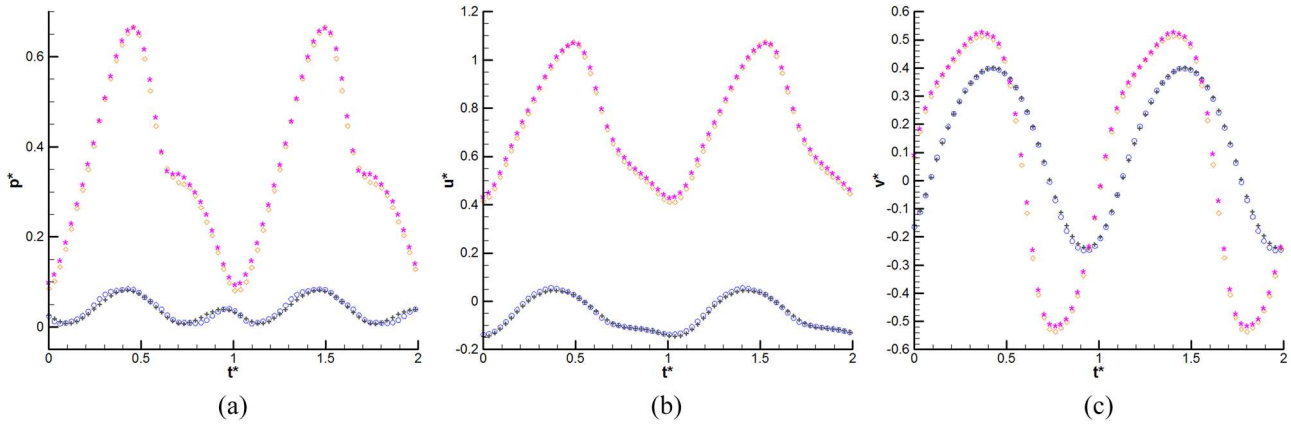

FIG. 11. Comparisons of velocity time histories in the wake flow between the model predictions and CFD results for Case 1: (a) pressure, (b) streamwise velocity, and (c) vertical velocity. Circle shape point, model prediction at point A; plus shape point, CFD results at point A; diamond shape point, model prediction at point B; and star shape point, CFD results at point B.

Published under license by AIP Publishing

regression tasks so that the dataset used for training and testing networks should be image like dataset. The flow field information value at each moment should be distributed over evenly distributed grid points, such as pixels.

We use high-precision numerical simulation to calculate the dynamic process of unsteady flow field and record the flow field information at each moment. Numerical simulations are conducted by solving the nondimensionalized incompressible Navier-Stokes equations as follows:

$$\nabla \cdot u = 0,$$
$$\frac{\partial u}{\partial t} + \nabla \cdot uu = -\nabla p + \frac{1}{Re}\nabla^2 u, \tag{5}$$

where u, x, y, t, and p are the nondimensionalized velocity, length, time, and pressure, respectively, by the incoming velocity U_0, the characteristic length D, the fluid density ρ, and vortex shedding frequency $1/T$. The finite volume method is used in the CFD solver. The Lower-Upper Symmetric Gauss-Seidel (LU-SGS) implicit method is employed for time integration. The second-order van Leer format is employed for spatial discretization. The two-dimensional laminar model is adopted for low Reynolds number flow. The three-dimensional Spalart-Allmaras (SA) turbulence model is adopted for high Reynolds number flow.

A rectangular area should be chosen as the sampling area. Lattice like sampling points of $Nx \times Ny$ are placed in the space, projecting the nondimensionalized flow field variables onto the uniformly distributed grid. The values of points inside the body are 0. Three-dimensional flow field variables (p^*, u^*, and v^*) are extracted at each sampling point. The $Nx \times Ny \times 3$ dimensional data that are extracted represent each instantaneous field. The obtained data are arranged in a chronological order to obtain a dataset for training and testing the neural network.

B. Training algorithm

The root mean square error (RMSE) is used to evaluate the model performance, i.e.,

$$\mathrm{RMSE}^t = \sqrt{\frac{\sum_{i=1}^{N}(\psi_i^t - \psi_{o,i}^t)^2}{N}}, \tag{6}$$

where ψ_i^t and $\psi_{o,i}^t$ denote the predictions and numerical simulations at the node i and the time level t, respectively, and N represents the number of nodes on the full image.

Training the network is equivalent to minimizing the loss function in Eq. (6) to obtain the optimal all kernel parameters. The backpropagation method is implemented for training, which involves calculating the gradients of the loss function with respect to the learnable parameters. In the backpropagation method, the sensitivity of every layer is the variable to back propagate the loss in Eq. (6).[34] Then, the gradients of the loss function with respect to the parameters of the network can be computed.

Adaptive moment estimation (Adam) is employed as the optimization algorithm to train the network.[35] In this algorithm, the exponential moving average is used to update the gradient vector and the squared gradient. Training of the network is carried out with the open-source software library TensorFlow.[36] The whole training procedure is as follows:

(1) Initialize network parameters, including weight W and offset b for each layer.
(2) Sample a batch in the training dataset, input = $\{F_1, F_2, \ldots, F_t\}$ and output = $\{F_{t+1}\}$.
(3) Update iteration step t + 1 flow field F_{t+1}.
(4) Compute the RMSE between F_{t+1} and F_{t+1} and get the gradient g_t of the loss function with respect to the parameters.
(5) Update network parameters, $W_{t+1} = W_t - \alpha f(g_t)$ and $b_{t+1} = b_t - \alpha f(g_t)$, where α is the learning rate.
(6) Repeat Steps 2–5, until the loss function converges.

IV. RESULTS AND DISCUSSIONS

A. Deep neural network training

We apply the hybrid deep neural network described in Sec. II on four representative experiments to illustrate the effectiveness of

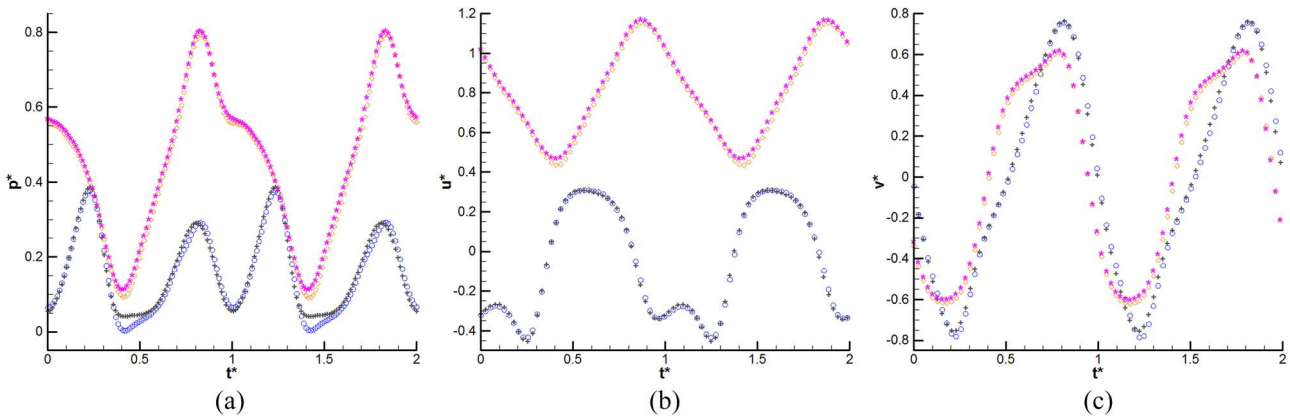

FIG. 12. Comparisons of velocity time histories in the wake flow between the model predictions and CFD results for Case 2: (a) pressure, (b) streamwise velocity, and (c) vertical velocity. Circle shape point, model prediction at point A; plus shape point, CFD results at point A; diamond shape point, model prediction at point B; and star shape point, CFD results at point B.

Published under license by AIP Publishing

the approach to nonlinear model reduction and predict the flow fields at future occasions based on flow fields at previous occasions. Numerical simulations, namely, flows past a cylinder at various Reynolds numbers and flow around an airfoil, were conducted. The first numerical experiment (Case 1) is a two-dimensional flow past a cylinder at Re = 200, and the second experiment (Case 2) is a three-dimensional flow past a cylinder at Re = 4000; the third experiment (Case 3) is a two-dimensional flow past the NACA0012 airfoil at Re = 8000 and angle of attack $\alpha = 20^\circ$. The fourth experiment (Case 4) is used to test the generalization ability of the neural network. In this experiment, two-dimensional flow field data at Re = 300, 400, 500, 600, 700, and 800 are employed as the dataset for training the hybrid deep neural network to capture the underlying dynamics in the Reynolds number range from 300 to 900, while the dataset for Re = 550 and 900 is used for testing. In all experiments, the flow field data were computed and processed to deep learning datasets by the same way.

Take the flow past a cylinder at Re = 200 experiment as example, detailing the construction process of the dataset. Flow is intended to be obtained by CFD calculation, and the computational area used is shown in Fig. 5. The distance from the center of the body to the inlet is 20D and to the outlet is 60D. The transverse

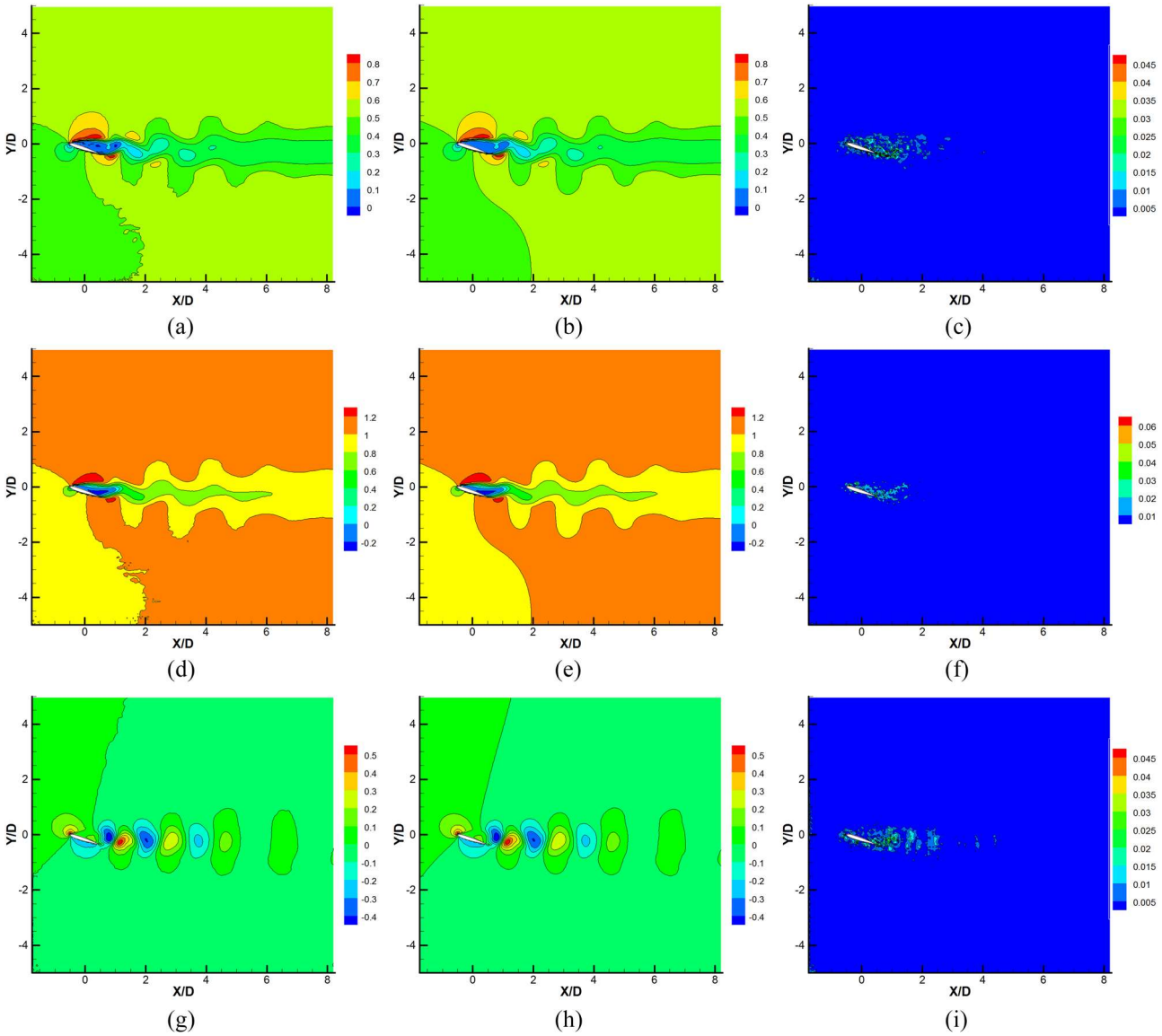

FIG. 13. Comparisons of instantaneous flow fields after 64 time-steps between the model predictions and CFD results for Case 3 (flow past an airfoil at Re = 8000): network predictions for (a) pressure, (d) streamwise velocity, and (g) vertical velocity; CFD results for (b) pressure, (e) streamwise velocity, and (h) vertical velocity; and absolute prediction error for (c) pressure, (f) streamwise velocity, and (i) vertical velocity.

Published under license by AIP Publishing

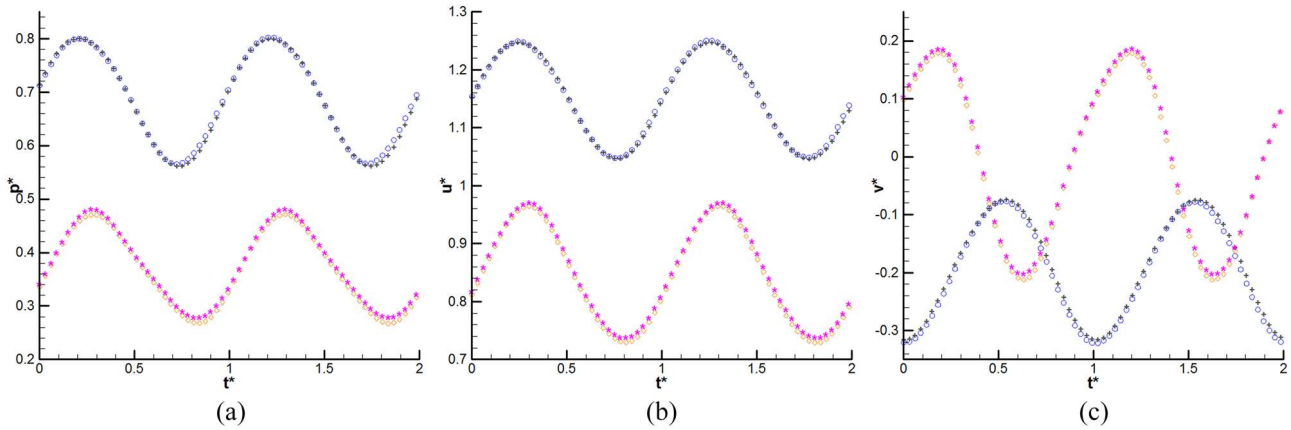

FIG. 14. Comparisons of velocity time histories in the wake flow between the model predictions and CFD results for Case 3: (a) pressure, (b) streamwise velocity, and (c) vertical velocity. Circle shape point, model prediction at point A; plus shape point, CFD results at point A; diamond shape point, model prediction at point B; and star shape point, CFD results at point B.

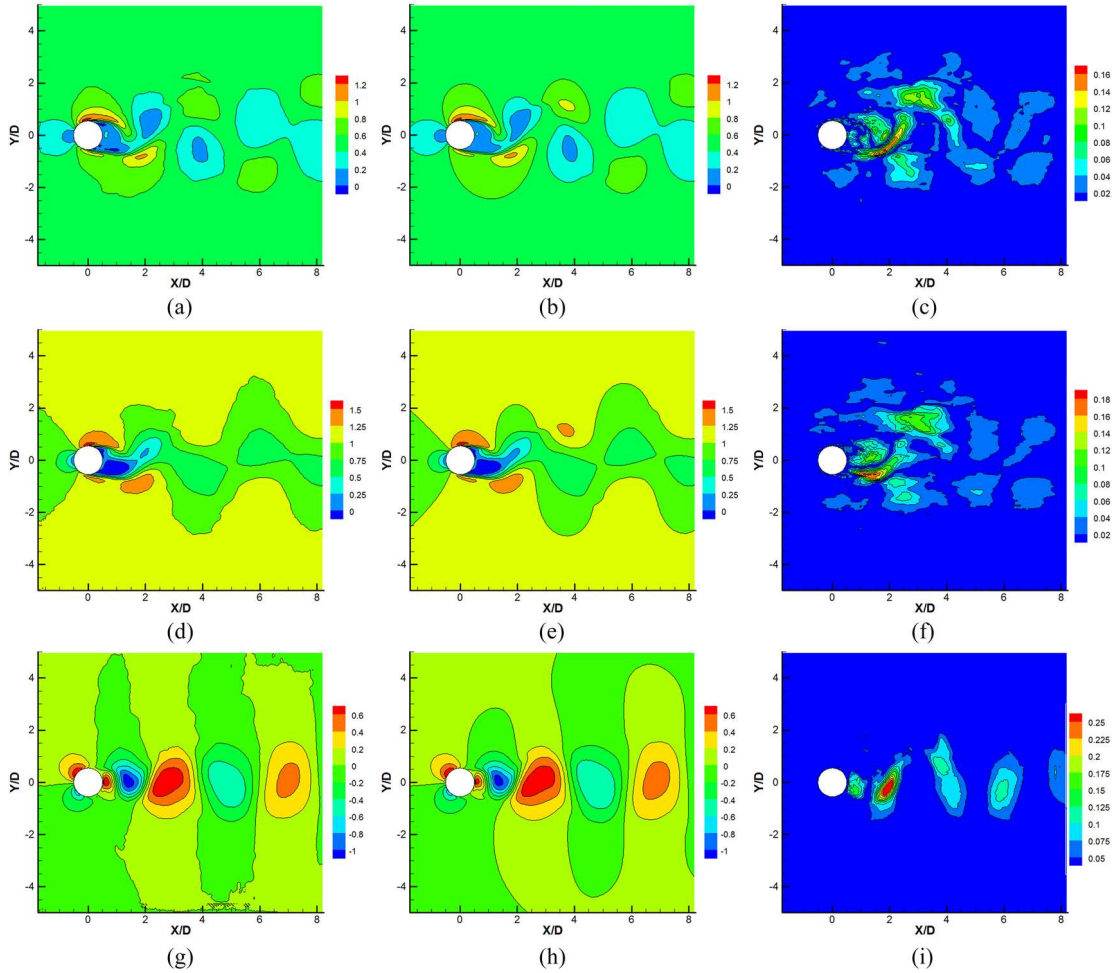

FIG. 15. Comparisons of instantaneous flow fields after 64 time-steps between the model predictions and CFD results for Case 4 (test results of flow past a cylinder at Re = 550): network predictions for (a) pressure, (d) streamwise velocity, and (g) vertical velocity; CFD results for (b) pressure, (e) streamwise velocity, and (h) vertical velocity; and absolute prediction error for (c) pressure, (f) streamwise velocity, and (i) vertical velocity.

Published under license by AIP Publishing

width of the computational domain is 80D. (The cross-flow direction domain sizes are 60D for three-dimensional flow.) A Dirichlet boundary condition is imposed on the inlet with the incoming velocity U_0; a free out-flow boundary condition is imposed on the outlet; a slip boundary condition is set up for the bottom and top boundaries of the flow; a no-slip boundary condition is set up for the solid body surface. (Symmetry boundary conditions are imposed on the two faces in the cross-flow direction for three-dimensional flow.) The grid resolution is tested to make the calculations accurate enough. After choosing the right grid, make sure there are about 64 step solutions in each flow cycle by adjusting the solving time step length.

We use a rectangular area as the sampling area (area inside the dotted line), as shown in Fig. 5. Lattice like sampling points of 200 × 200 are placed in the space, shown in Fig. 6. Then, project the nondimensionalized flow field variable quantities onto the uniformly distributed grid. The values of the point inside the body are 0. Three-dimensional flow field variables (p, u, and v) are extracted at each sampling point. The 200 × 200 × 3 dimensional data that are extracted represent each instantaneous field. The obtained data are arranged in chronological order to obtain a dataset for training and testing the neural network. Each dataset includes 10 000 time-step flow field data, of which 9500 time-steps are used as training sets and 500 time-steps are used as test sets.

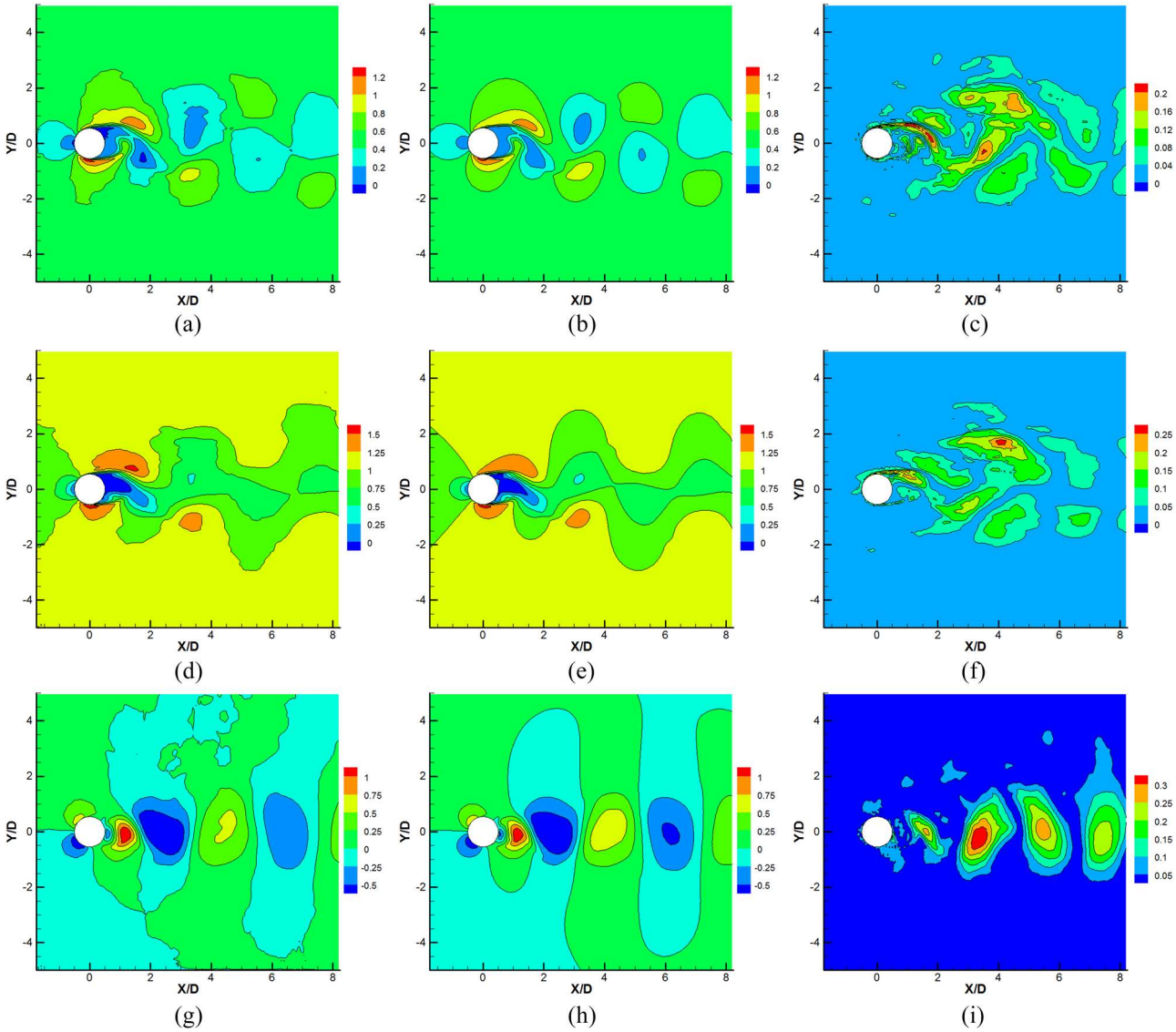

FIG. 16. Comparisons of instantaneous flow fields after 64 time-steps between the model predictions and CFD results for Case 4 (test results of flow past a cylinder at Re = 900): network predictions for (a) pressure, (d) streamwise velocity, and (g) vertical velocity; CFD results for (b) pressure, (e) streamwise velocity, and (h) vertical velocity; and absolute prediction error for (c) pressure, (f) streamwise velocity, and (i) vertical velocity.

Published under license by AIP Publishing

Then, we train the proposed hybrid deep neural networks in Tensorflow, with the values of the parameters for each layer, as listed in Table I. The learning rate used in optimization is set as 0.0005. Figure 7 shows that the training error defined by Eq. (6) for four datasets decreases with increasing number of training steps. After 700 training epochs, the training error converges to less than 0.02.

Using the trained network, a flow field at time step i is recursively predicted based on the set of input flow field images at time step i-1 to i-16. After the CNN layers, the size of flow fields reduced from 200 × 200 × 3 to 7 × 7 × 64. The original flow field sample size is 38 times the sample size after the reduced order. To achieve the goal of predicting the flow fields at future occasions based on flow fields at previous occasions continuously, we recycle the output of the trained network to its input and updated the input recursively as the time-step advancement, with an initial condition taken from 16 snapshots of CFD data, so that the network is able to achieve long-time predictions of flow fields even without known CFD data in the coming period. The trained neural network could get 500 steps flow fields in 74 s, while CFD code needs at least 830 s to complete the same simulation. The flow field prediction model proposed in this paper is 11 times faster than the traditional method.

B. The flow around a cylinder

We use the first two experiments to test the ability of the neural networks to predict unsteady flow in laminar and turbulent and reveal whether the neural network can capture spatial-temporal features of the unsteady flow at different states. The first numerical experiment (Case 1), the flow past a cylinder at Re = 200, is a laminar flow; and the second experiment (Case 2), the flow past a cylinder at Re = 4000, is a turbulent flow. They have different physical characteristics. Whether the same neural network structure can be applied to two different flow fields is the criterion for testing the applicability of the hybrid deep neural network. The RMSE between the predicted and accurate results of the flow fields in the test phase for those two cases are less than one percent during two cycles.

Comparisons of instantaneous flow fields between the hybrid deep network predicted results and CFD results after 64 time-steps for two cases are shown in Figs. 8 and 9. From comparisons, we can get that flow fields predicted are found to agree well with CFD simulation flow fields for all cases. Two characteristic positions, shown in Fig. 10, are selected to show the time series prediction accuracy. Time series of three flow field variables at these positions, predicted by the network and calculated by CFD, are compared in Figs. 11 and 12. All predicted results agree very well with CFD results. It is proved that each part of the neural network structure has completed the predetermined target and realized the prediction of the unsteady flow fields. The neural network can not only capture the evolution process of the flow fields but also represent the low-dimensional features to larger dimensional accurate representation. Thus, we can conclude that this kind of hybrid deep neural network can predict the spatial-temporal evolution of laminar and turbulent flow.

C. The flow around an airfoil

In order to further explore the applicability of the neural network structure, the neural network is used to predict the more complex unsteady flow fields, the flow around an airfoil at a higher Reynolds number. The RMSE between the predicted and accurate results of the flow fields in the test phase for this case is less than one percent during two cycles. Comparisons of instantaneous flow fields between the hybrid deep network predicted results and CFD results are shown in Fig. 13. The time series of three flow field variables at two selected positions, predicted by the model and calculated by the CFD solver, respectively, are compared in Fig. 14. All predicted results agree very well with those of the CFD solver. The error accumulation on differences between the predicted and the CFD simulation flow fields as the number of the recursive step increases is not obvious. We can get that this kind of hybrid deep neural network can predict the spatial and temporal evolution of turbulence flow over a complex shape (airfoil). We treat the input-output as images, of which the different characteristics are easy to be captured by the CNN layer, so that this hybrid deep neural network is suitable for learning the spatial-temporal features of different unsteady flows.

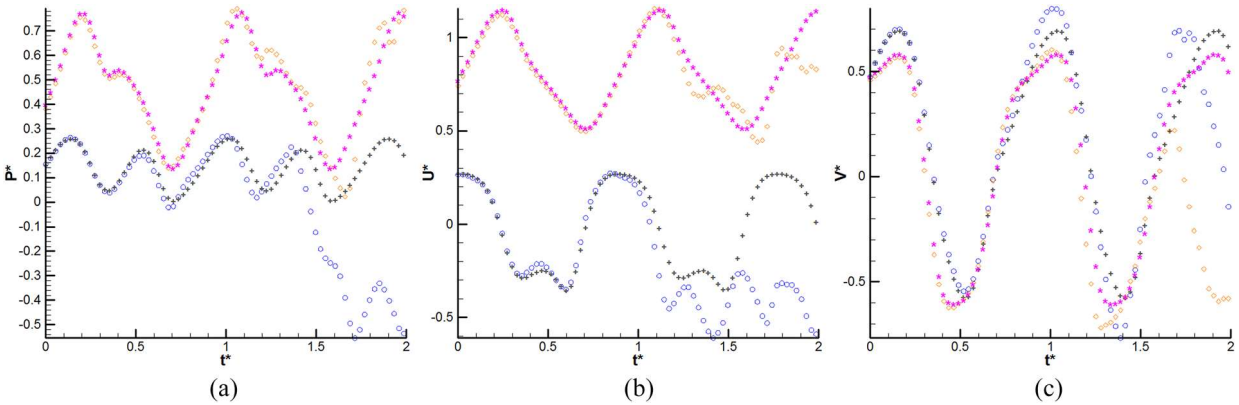

FIG. 17. Comparisons of velocity time histories in the wake flow between the model predictions and CFD results for Case 4 (test results of flow past a cylinder at Re = 550): (a) pressure, (b) streamwise velocity, and (c) vertical velocity. Circle shape point, model prediction at point A; plus shape point, CFD results at point A; diamond shape point, model prediction at point B; and star shape point, CFD results at point B.

Published under license by AIP Publishing

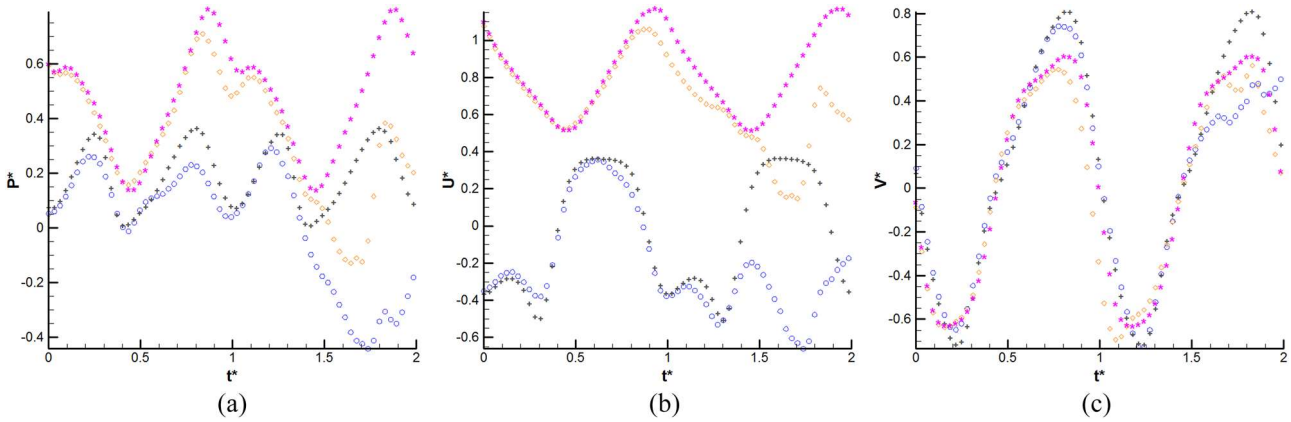

FIG. 18. Comparisons of velocity time histories in the wake flow between the model predictions and CFD results for Case 4 (test results of flow past a cylinder at Re = 900): (a) pressure, (b) streamwise velocity, and (c) vertical velocity. Circle shape point, model prediction at point A; plus shape point, CFD results at point A; diamond shape point, model prediction at point B; and star shape point, CFD results at point B.

D. The flows at different Reynolds numbers

The generalization ability refers to the ability of machine learning algorithms to adapt to new data. It is generally expected that a network trained with training samples will have a strong generalization ability, that is, the ability to give a reasonable response to new inputs. The fourth experiment is used to test the generalization ability of the neural network, whether the hybrid deep neural network could be used to predict the flow field structures that the trained networks have not meet. In this experiment, the flow field data at Re = 300, 400, 500, 600, 700, and 800 are employed as the dataset for training the hybrid deep neural network to capture the underlying dynamics, while the dataset for Re = 550 and 900 is used for testing. The RMSE between the predicted and accurate results of the flow fields in the test phase for those two prediction cases are less than five percent during the first prediction cycle.

Comparisons of instantaneous flow fields between the hybrid deep network predicted results and CFD results after 64 time-steps for two test cases are shown in Figs. 15 and 16. Time series of the three flow field variables at selected positions, predicted by the network and calculated by CFD, for two test cases are compared in Figs. 17 and 18. From comparisons, we can get that the flow fields predicted by the network still show good agreement with the CFD results in 64 time-steps, even though the trained networks have not been trained with the flow structures at the tested Reynolds number. Even though the absolute prediction error is higher than former cases, the flow field structural features are well predicted. It means that the Reynolds number effects have also been learned by the hybrid deep neural network. The results show that the hybrid deep neural network has good generalization ability at least in one cycle.

V. CONCLUSIONS

A novel hybrid deep neural network architecture was designed to capture the spatial-temporal features directly from high-dimensional unsteady flow fields. The hybrid deep neural network is constituted by CNN, ConvLSTM, and DeCNN. The CNN layers are designed to capture the complex mapping directly from high-dimensional input flow fields and represent it in a low-dimensional form. The ConvLSTM layer is designed to capture temporal features between low-dimensional feature representations and predict the low-dimensional features of flow fields at future occasions. The DeCNN layers are used to represent the predicted low-dimensional features to high-dimensional output fields, with the same dimension as input fields. This kind of deep neural network can capture accurate spatial-temporal information from the spatial-temporal series of unsteady flows.

The flow around a cylinder at various Reynolds numbers and the flow around an airfoil at higher Reynolds number are carried out to establish the datasets training the networks separately. The trained hybrid deep neural networks were then tested by the prediction of the flow fields at future occasions. The predicted flow fields using the trained hybrid deep neural networks are in good agreement with the flow fields calculated directly by the computational fluid dynamic solver. The Reynolds number effects have also been learned by the hybrid deep neural network, which shows that the proposed method has good generalization capability. This hybrid deep neural network can achieve fast and accurate prediction of unsteady flow fields, which is very important to flow control and aerodynamic optimization application.

All the test cases show that the maximum errors are located accelerating boundary layers on the cylinder wall or in the wake, where the flow field changes drastically, but the maximum errors are in a low level. The prediction accuracy of the hybrid deep neural networks is acceptable. This study shows the potential capability of the novel hybrid deep neural network based reduced order model in the fast prediction of the unsteady flow. The new prediction method can be used in fluid-structure interactions and flow control, where the fast high-dimensional nonlinear unsteady flow calculation is needed.

The proposed hybrid deep neural network shows good potential in spatial-temporal flow features modeling; however, there is still much work to be done. The ultimate goal of the DNN based reduced

Published under license by AIP Publishing

order model is to use the hybrid deep neural network to predict complex turbulence flow fields in different Reynolds numbers and the highly unsteady flows with moving boundaries. In the next step, we will further optimize the network structure and find more new flow features to tagging the flow training data so that it can predict the unsteady flows at different Reynolds numbers in a longer period with more accuracy. The proposed new method is also expected to be used in the fluid-structure interactions and flow control in future.

ACKNOWLEDGMENTS

This work was partially supported by the National Natural Science Foundation of China (Grant Nos. 11872293 and 11672225), the Key Laboratory of Aerodynamics Noise Control (Grant No. 1801ANCL20180103), the Program of Introducing Talents of Discipline to Universities (known as the "111" Program Grant No. B18040), and the Key Laboratory of Reliability and Environment Engineering (No. 6142004190307).

REFERENCES

[1]F. Fang, C. Pain, I. M. Navon, A. H. Elsheikh, J. Du, and D. Xiao, "Non-linear Petrov-Galerkin methods for reduced order hyperbolic equations and discontinuous finite element methods," J. Comput. Phys. **234**, 540–559 (2013).
[2]D. J. Lucia, P. S. Beran, and W. A. Silva, "Reduced-order modeling: New approaches for computational physics," Prog. Aerosp. Sci. **40**, 51–117 (2004).
[3]E. H. Dowell, "Eigen mode analysis in unsteady aerodynamics: Reduced-order models," Appl. Mech. Rev. **50**(6), 371 (1997).
[4]P. J. Schmid, "Dynamic mode decomposition of numerical experimental data," J. Fluid Mech. **656**(10), 5–28 (2010).
[5]M. J. Henshawa, K. J. Badcock, and G. A. Vio, "Non-linear aeroelastic prediction for aircraft applications," Prog. Aerosp. Sci. **43**(4-6), 65–137 (2007).
[6]G. Chen, D. Li, Q. Zhou, A. Da Ronch, and Y. Li, "Efficient aeroelastic reduced order model with global structural modifications," Aerosp. Sci. Technol. **76**, 1–13 (2018).
[7]M. R. Jovanovie, P. J. Schmid, and J. W. Nichols, "Sparsity-promoting dynamic mode decomposition," Phys. Fluids **26**(2), 024103 (2014).
[8]M. S. Hemati, M. O. Williams, and C. W. Rowley, "Dynamic mode decomposition for large and streaming datasets," Phys. Fluids **26**(11), 111701 (2014).
[9]Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature **521**(7553), 436–444 (2015).
[10]I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016).
[11]T. N. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, 2013), pp. 8614–8618.
[12]H. Y. Xiong, B. Alipanahi, L. J. Lee, H. Bretschneider, D. Merico, R. K. Yuen, Y. Hua, S. Gueroussov, H. S. Najafabadi, and T. R. Hughes, "The human splicing code reveals new insights into the genetic determinants of disease," Science **347**(6218), 1254806 (2015).
[13]J. Ling, A. Kurzawski, and J. Templeton, "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance," J. Fluid Mech. **807**, 155–166 (2016).
[14]J. Wu, H. Xiao, and E. Paterson, "Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework," Phys. Rev. Fluids **3**, 074602 (2018).
[15]R. Maulik, O. San, A. Rasheed, and P. Vedula, "Subgrid modelling for two-dimensional turbulence using neural networks," J. Fluid Mech. **858**, 122–144 (2019).
[16]C. Xie, J. Wang, H. Li, M. Wan, and S. Chen, "Artificial neural network mixed model for large eddy simulation of compressible isotropic turbulence," Phys. Fluids **31**, 085112 (2019).
[17]M. Wang, H. Li, X. Chen, and Y. Chen, "Deep learning-based model reduction for distributed parameter systems," IEEE Trans. Syst., Man, Cybern.: Syst. **46**(12), 1664–1674 (2016).
[18]N. Omataa and S. Shirayama, "A novel method of low-dimensional representation for temporal behavior of flow fields using deep autoencoder," AIP Adv. **9**, 015006 (2019).
[19]K. Fukami, K. Fukagata, and K. Taira, "Super-resolution reconstruction of turbulent flows with machine learning," J. Fluid Mech. **870**, 106–120 (2019).
[20]X. Jin, P. Cheng, W. Chen, and H. Li, "Prediction model of velocity field around circular cylinder over various Reynolds numbers by fusion convolutional neural networks based on pressure on the cylinder," Phys. Fluids **30**, 047105 (2018).
[21]V. Sekar, Q. Jiang, C. Shu, and B. Khoo, "Fast flow field prediction over airfoils using deep learning approach," Phys. Fluids **31**, 057103 (2019).
[22]A. T. Mohan and D. V. Gaitonde, "A deep learning based approach to reduced order modeling for turbulent flow control using LSTM neural networks," preprint arXiv:1804.09269 (2018).
[23]S. Pawar, S. M. Rahman, H. Vaddireddy, O. San, A. Rasheed, and P. Vedula, "A deep learning enabler for nonintrusive reduced order modeling of fluid flows," Phys. Fluids **31**, 085101 (2019).
[24]Z. Deng, Y. Chen, Y. Liu, and K. Kim, "Time-resolved turbulent velocity field reconstruction using a long short-term memory (LSTM)-based artificial intelligence framework," Phys. Fluids **31**, 075108 (2019).
[25]A. T. Mohan, D. Daniel, M. Chertkov, and D. Livescu, "Compressed convolutional LSTM: An efficient deep learning framework to model high fidelity 3D turbulence," preprint arXiv:1903.00033 (2019).
[26]S. Lee and D. You, "Data-driven prediction of unsteady flow over a circular cylinder using deep learning," J. Fluid Mech. **879**, 217–254 (2019).
[27]M. Ruttgers, S. Lee, and D. You, "Prediction of typhoon tracks using a generative adversarial network with observational and meteorological data," Sci. Rep. **9**, 6057 (2019).
[28]A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Neural Information Processing Systems Conference* (2012), Vol. 25, pp. 1097–1105.
[29]V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *International Conference on International Conference on Machine Learning* (Omnipress, 2010), Vol. 27, pp. 807–814.
[30]F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," Neural Comput. **12**(10), 2451–2471 (2000).
[31]X. Shi, Z. Chen, H. Wang, and D. Yeung, "Convolutional LSTM network: A machine learning approach for precipitation now casting," in *Neural Information Processing Systems Conference* (2015), Vol. 28, pp. 802–810.
[32]M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (IEEE, 2010).
[33]M. D. Zeiler, D. Matthew, G. W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *2011 International Conference on Computer Vision* (IEEE, 2012).
[34]J. Bouvrie, *Notes on Convolutional Neural Networks* (Massachusetts Institute of Technology, 2006).
[35]D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," presented at the Third International Conference on Learning Representations, San Diego, CA, USA, 2015; e-print arXiv:1412.6980.
[36]M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, R. Jozefowicz, Y. Jia, M. Kudlur, L. Kaiser, J. Levenberg, D. Mane, M. Schuster, S. Moore, R. Monga, D. Murray, C. Olah, J. Shlens, I. Sutskever, B. Steiner, K. Talwar, P. Tucker, V. Vasudevan, V. Vanhoucke, F. Viegas, P. Warden, O. Vinyals, M. Wattenberg, M. Wicke, X. Zheng, and Y. Yu, "TensorFlow: Large-scale machine learning on heterogeneous systems," Software available at http://tensorflow.org/, 2015.

Published under license by AIP Publishing