Physics of Fluids

# Deep learning methods for super-resolution reconstruction of turbulent flows

Cite as: Phys. Fluids **32**, 025105 (2020); https://doi.org/10.1063/1.5140772
Submitted: 01 December 2019 . Accepted: 21 January 2020 . Published Online: 12 February 2020

Bo Liu, Jiupeng Tang, Haibo Huang, and Xi-Yun Lu

Physics of Fluids ARTICLE scitation.org/journal/phf

# Deep learning methods for super-resolution reconstruction of turbulent flows

Cite as: Phys. Fluids 32, 025105 (2020); doi: 10.1063/1.5140772
Submitted: 1 December 2019 • Accepted: 21 January 2020 •
Published Online: 12 February 2020



Bo Liu, Jiupeng Tang, Haibo Huang,[a] and Xi-Yun Lu

## AFFILIATIONS

Department of Modern Mechanics, University of Science and Technology of China, Hefei 230026, China

[a]Author to whom correspondence should be addressed: huanghb@ustc.edu.cn

## ABSTRACT

Two deep learning (DL) models addressing the super-resolution (SR) reconstruction of turbulent flows from low-resolution coarse flow field data are developed. One is the static convolutional neural network (SCNN), and the other is the novel multiple temporal paths convolutional neural network (MTPC). The SCNN model takes instantaneous snapshots as an input, while the MTPC model takes a time series of velocity fields as an input, and it includes spatial and temporal information simultaneously. Three temporal paths are designed in the MTPC to fully capture features in different time ranges. A weight path is added to generate pixel-level weight maps of each temporal path. These models were first applied to forced isotropic turbulence. The corresponding high-resolution flow fields were reconstructed with high accuracy. The MTPC seems to be able to reproduce many important features as well, such as kinetic energy spectra and the joint probability density function of the second and third invariants of the velocity gradient tensor. As a further evaluation, the SR reconstruction of anisotropic channel flow with the DL models was performed. The SCNN and MTPC remarkably improve the spatial resolution in various wall regions and potentially grasp all the anisotropic turbulent properties. It is also shown that the MTPC supplements more under-resolved details than the SCNN. The success is attributed to the fact that the MTPC can extract extra temporal information from consecutive fluid fields. The present work may contribute to the development of the subgrid-scale model in computational fluid dynamics and enrich the application of SR technology in fluid mechanics.



## NOMENCLATURE

| | |
|---|---|
| $C$ | the number of channels of each sample |
| $F$ | functional mapping |
| $H$ | the height of each sample |
| $n_{snapshots}$ | the number of snapshots |
| $Q$ | the second invariant of the velocity gradient tensor |
| $r$ | upscaling factor |
| $R$ | the third invariant of the velocity gradient tensor |
| $W$ | the width of each sample |
| $y^+$ | distance from the wall normalized by the viscous length scale |

### Greek letters

| | |
|---|---|
| $\varepsilon$ | $L_2$ error norms |
| $\rho$ | correlation coefficient |
| $\theta$ | model parameters |

### Abbreviations

| | |
|---|---|
| CFD | computational fluid dynamics |
| CNN | convolutional neural network |
| DL: | deep learning |
| DNS | direct numerical simulation |
| EMSRB | enhanced multi-scale residual block |
| HR | high-resolution |
| JHTDB | Johns Hopkins turbulence databases |
| LES | large eddy simulation |
| LR | low-resolution |
| MTPC | multiple temporal paths convolutional neural network |
| PDF | probability density function |
| PIV | particle image velocimetry |
| ReLU | rectified linear unit |
| SCNN | static convolutional neural network |
| SR | super-resolution |

## I. INTRODUCTION

With the advances of high-performance computers and experimental techniques, it is easier for researchers to get high-resolution (HR) fluid data. Experimental flow field measurements, e.g., particle image velocimetry (PIV),[1] could capture the intricate details of turbulent flows. We are also able to obtain high-resolution flow fields using computational fluid dynamics (CFD) methods, such as large eddy simulation (LES) and direct numerical simulation (DNS). However, in industry, we may expect to achieve high-resolution (HR) flow data quickly with less effort, but experiments and simulations still seem to be expensive and time-consuming.[2] It becomes essential to develop an efficient way to obtain the HR data. Because experimental and CFD results for turbulence are data-rich, the rapidly developed data-driven technology may provide an efficient way to generate high-resolution flow field quickly without solving partial differential equations again and again. In this study, we focus on reconstructing turbulent flows from spatiotemporal low-resolution (LR) data through deep learning (DL) technology.

Deep learning, as an important branch of machine learning, has been widely applied in fluid mechanics in the past few years.[3–7] Ling *et al.*[8] proposed a novel deep neural network architecture embedded with Galilean invariance to model the anisotropic Reynolds stresses. Their results have shown the superiority of combining deep learning with domain knowledge. Tracey *et al.*[9] used a feed-forward neural network to model the source terms in the Spalart–Allmaras turbulence model. In their work, DL served as a new way to develop turbulence models. Besides, the convolutional neural network (CNN) is a powerful tool to extract fluid dynamics features and predict flow fields. Jin *et al.*[10] designed a fusion CNN to predict the velocity field around a circular cylinder solely based on pressure fluctuations on the cylinder, which are readily available in fluid mechanics experiments. Apart from that, deep reinforcement learning has been successfully applied in the studies of flow control[11,12] and collective motion of creatures.[13,14] Deep learning technology has become a promising tool in these fields.

Super-resolution (SR) is a class of techniques that improve the resolution of an imaging system. As a consequence of imperfections of measuring devices or instability of the observed scene, the attained images may be noisy and flawed in spatial and temporal resolution.[15] The SR is able to leverage high-resolution data on a smaller domain to enhance the resolution on a larger imaging system.[16] The deep learning method was first used in the SR problem by Dong *et al.*[17] in 2014. They proposed a model termed SRCNN and showed enormous potentials of the learning method compared with traditional methods such as sparsity-based techniques[18,19] and bicubic interpolation. Thenceforth, massive novel deep learning models, such as FSRCNN,[20] LapSRN,[21] and DBPN,[22] have been designed. The SR is becoming a more attractive research area in the computer vision field.

Similarly, it is quite common to encounter handicapped data resulting from the limitation of sensor resolution in experiments or insufficient mesh size in the CFD simulations. In the CFD, mesh refinement is a direct way to get HR data, but the calculation cost of numerical simulation may grow exponentially. A typical DNS may take a couple of days. For the experiment, improving the spatial resolution of the equipment is the only way to get higher quality data. However, it may bring expensive additional expenses. On the other hand, deep learning methods are much more efficient and economical. The computational time is only relevant to the size of the networks. Once the networks have been trained, the time spent on calculating is usually about several seconds.

Benefiting from the mushroom growth of deep learning technology, super-resolution has been developed in computational fluid dynamics recently. Fukaimi *et al.*[23] proposed a hybrid Downsampled Skip-Connection Multi-Scale (DSC/MS) model to reconstruct the flow field of two-dimensional decaying isotropic turbulence. The model could enhance the spatial resolution of velocity field greatly even with only 50 training samples. The kinetic energy spectra could also be accurately preserved. However, the studied case is simple, and it is relatively easy to grasp the flow characteristics of two-dimensional isotropic turbulence. In the field of experimental fluid dynamics, many efforts have been deployed in improving the spatial resolution of PIV data.[24,25] To better extract PIV velocity from the particle image patches, the PIV-DCNN model with four-level regression deep convolutional neural networks was developed.[26] Although the fine structures of natural complex flow are extracted, the computational efficiency is low. Deng *et al.*[27] used two generative adversarial network-based models, SRGAN and ESRGAN, to augment the spatial resolution of complicated wake flow behind two side-by-side cylinders. Their result shows that the SR technology works well even in an intricate flow configuration.

It is noticed that all the models mentioned above, e.g., DSC/MS, only utilize the spatial information on the low-resolution (LR) fluid field at an instant $I_t^{LR}$ to generate the corresponding HR fluid field at the instant $I_t^{HR}$. We call these kinds of models "static models." Static models try to establish a mapping $F$ from LR data to HR data, i.e., $I_t^{HR} = F(I_t^{LR}, \theta)$, where $\theta$ are the parameters of the models. As we know, turbulent flows are time-space coupled. Therefore, we wonder if the supplement of temporal information could help to further improve the quality of the reconstruction. To incorporate the idea, we developed a novel multiple temporal paths convolutional neural network (MTPC). Unlike the static models, MTPC takes a sequence of low-resolution data $I_{[t-d:t+d]}^{LR}$ as an input. The network can readily extract temporal information from the consecutive snapshots. In this way, the spatiotemporal information could be considered simultaneously, and the hidden physical mechanism may be revealed. In other words, the MTPC tries to establish a mapping function like $I_t^{HR} = F(I_{[t-d:t+d]}^{LR}, \theta)$. We would check whether the MTPC is able to reconstruct better results.

The rest of this paper is structured as follows. Section II introduces the basic knowledge of the convolutional neural network and the proposed model framework. Section III describes the dataset used in this work. In Sec. IV, the results of two test cases are presented. One is about forced isotropic turbulence, and the other is turbulent channel flow. Finally, conclusions and outlook are presented in Sec. V.

## II. METHOD

The purpose of this work is to reconstruct high-resolution turbulent fluid data $I^{HR}$ from a sequence of low-resolution data $I_{[t-d:t+d]}^{LR}$. These LR data are obtained by filtering the high-resolution data $I_{[t-d:t+d]}^{HR}$ down using a box filter. Our objective is to learn an

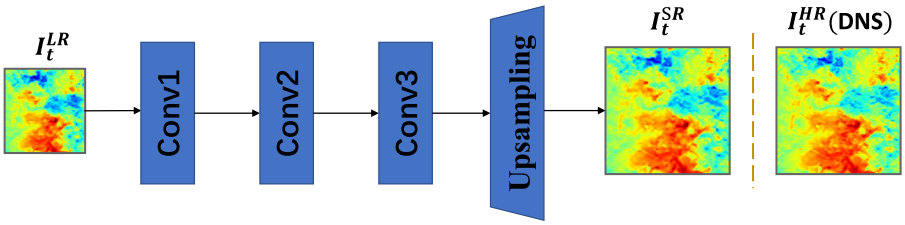


**FIG. 1**. Schematic diagram of the static convolutional neural network (SCNN). It takes instantaneous low-resolution snapshots $I_t^{LR}$ as an input and generates the corresponding high-resolution fluid field $I_t^{HR}$.

end-to-end mapping function $F$ to predict the HR flow field,

$$I^{SR} = F(I^{LR}_{[t-d:t+d]}, \theta), \tag{1}$$

where $I^{SR}$ is the SR reconstructed result and $\theta$ denote all the learnable parameters of CNN. In the case of $d = 0$, the mapping function becomes $I^{SR} = F(I_t^{LR}, \theta)$, which is the usual single-input–single-output case. In this situation, the input is only instantaneous spatial data without temporal information. It should be noted that the single image super-resolution is an ill-posed problem because one input LR data may lead to several different high-resolution outputs. The solution to the super-resolution problem relies on data redundancy provided by training pairs. $I^{HR}$ can surely be super-resolved independently as the single-input case, but it is not wise to waste the details available from the other LR frames.[22] In the multiple-input–single-output case, the shared explicit redundancy from adjacent frames would provide more constrains, thus alleviating the ill-posedness of such an inverse problem.

Suppose a training set contains some instances, e.g., $\{I_t^{HR}, I^{LR}_{[t-d:t+d]}\}$, the training process of machine learning is equivalent to solving an optimization problem,

$$\theta = \arg\min_{\theta} \mathscr{L}(F(I^{LR}_{[t-d:t+d]}, \theta), I^{HR}), \tag{2}$$

where $\mathscr{L}$ denotes the loss function. Once the training process is finished, $\theta$ is fixed for the test.

Two deep learning models are tested in the present study. They are the static convolutional neural network (SCNN) taking instantaneous spatial data as an input and the multiple temporal paths convolutional neural network (MTPC) taking a sequence of data as an input. We will present more details about the two models in Secs. II A and II B .

### A. Static convolutional neural network

Convolutional Neural Networks, as powerful tools in deep learning, have been widely used in image classification,[28] object detection,[29] image segmentation,[30] and many other fields. For a network with $L$ layers, each layer's input comes from the output of its former layer. The procedure can be described as

$$f^l = \phi(w_l * f^{l-1} + b_l), l = 1, \ldots, L, \tag{3}$$

with $f^0 = I^{LR}$, where $f^l$ is the input feature maps of layer $l+1$, and $w_l$, $b_l$ are the learnable weights and biases of the network, respectively. $\phi$ is the nonlinear activation function and due to $\phi$, the network is able to approximate an arbitrary complex function. Two of the most widely used activation functions in regression problem are the rectified linear unit (ReLU) $\phi(x) = \max(x, 0)$ and the tanh activate function $\phi(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$.

The architecture of SCNN is shown in Fig. 1, and the input is instantaneous snapshots of fluid data. Suppose $H$, $W$, and $C$ denote the height, width, and channels of the image, respectively, the shape of a two-dimensional (2D) snapshot is $H \times W \times C$. In this architecture, the input frame is first transmitted to three consecutive convolutional layers for feature detection and field reconstruction. After that, the data size is scaled to $rH \times rW \times C$ through an upsampling operator, where $r$ is the upscaling factor. There are several up-sampling methods available in the literature to recover the resolution from down-sampling layers.[15] One of the most commonly used methods is to upscale an input image to HR space using bicubic interpolation at the beginning of networks, which means that the SR operator is performed in HR space. Here, in the SCNN, an efficient sub-pixel convolution layer which was first proposed by Shi[31] is adopted. This layer, also termed PixelShuffle, transforms the shape of an $H \times W \times C \times r^2$ tensor into $rH \times rW \times C$ using a periodic shuffling operator shown in Fig. 2 at the end of the network. Thanks to PixelShuffle, feature extraction can be performed in LR space instead of HR space, thus making the SR process more efficient.

The square of the $L_2$ error loss function is chosen as the objective optimization function, which can be written as

$$\mathscr{L}_2(I^{SR}, I^{HR}) = \frac{1}{N}\sum_{t=1}^{N} \left\| F(I_t^{LR}; \theta) - I_t^{HR} \right\|_2^2. \tag{4}$$

Details of the parameters in SCNN are listed in Table I.

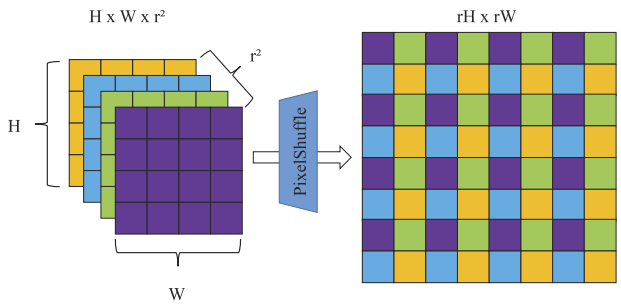


**FIG. 2**. Schematic diagram of the PixelShuffle. It transforms an $H \times W \times C \times r^2$ tensor into a tensor of shape $rH \times rW \times C$.

**TABLE I**. Detailed structure of the static convolutional neural network (SCNN).

| Layer | Kernel size/filters | Activation function | Output size |
|---|---|---|---|
| Input | ... | ... | $H \times W \times C$ |
| Conv1 | $5 \times 5/64$ | tanh | $H \times W \times 64$ |
| Conv2 | $5 \times 5/64$ | tanh | $H \times W \times 64$ |
| Conv3 | $5 \times 5/C \times r^2$ | tanh | $H \times W \times C \times r^2$ |
| Upsampling | ... | ... | $rH \times rW \times C$ |

## B. Multiple temporal paths CNN

Static models such as SCNN take one instantaneous low-resolution snapshot as an input. These models can only utilize spatial information and completely ignore the dynamic characteristics of turbulence. Actually, the turbulent processes are non-local in both time and space. In other words, the turbulent behavior at a point may be influenced by the flow remote from that point.[32] The MTPC has an advantage of taking a temporal sequence of low-resolution snapshots as an input. It can observe the turbulent motions through consecutive snapshots and thus grasp the temporally non-local properties as well as spatially non-local properties. The integration of spatiotemporal information may improve the reconstruction of the fine-scale structure of turbulence. The detailed description of the MTPC is presented in Subsections II B 1–II B 4.

### 1. Architecture

The MTPC takes a temporal sequence of LR snapshots $I^{LR}_{[t-d:t+d]}$ as an input. Figure 3 shows the architecture of the MTPC. It is shown that the network is composed of three temporal paths for prediction and one weight path to determine the weights of each predicted result. The three temporal paths are termed Backward Temporal Path (BTP), Central Temporal Path (CTP), and Forward Temporal Path (FTP), respectively, according to their inputs. The BTP takes $I^{LR}_{[t-d:t]}$ as its input. This branch outputs a potential HR result $I^{BT}_t$ by observing frames at the current instant $t$ and the former. The BTP looks like the idea of the backward temporal differential method in the CFD. Correspondingly, the CTP and FTP have some similarity to the central and forward temporal differential methods, respectively. Through observing different motion ranges, the three paths can extract diverse temporal information.

### 2. Temporal path

Each temporal path consists of a feature extraction module and a reconstruction module. The feature extraction module is composed of a convolutional layer and two basic blocks. The exploited features from the first layer and basic blocks are concatenated through skip connections. In this way, the possible disappearance of features during transmission is avoided. The basic block utilized here is the enhanced multi-scale residual block (EMSRB) shown in Fig. 4. The EMSRB is developed from the multi-scale residual block (MSRB),[33] which has been shown as an efficient block to detect image features at different scales. A novel two-bypass network with different sized convolutional kernels is used in the EMSRB to extract flow characteristics. The extracted features are concatenated and sent to a $1 \times 1$ convolutional layer to reduce the number of feature maps. We modified the MSRB by adding a $1 \times 1$ convolutional layer without any following activation functions in the residual path. With this modification, the model is able to adjust the residual

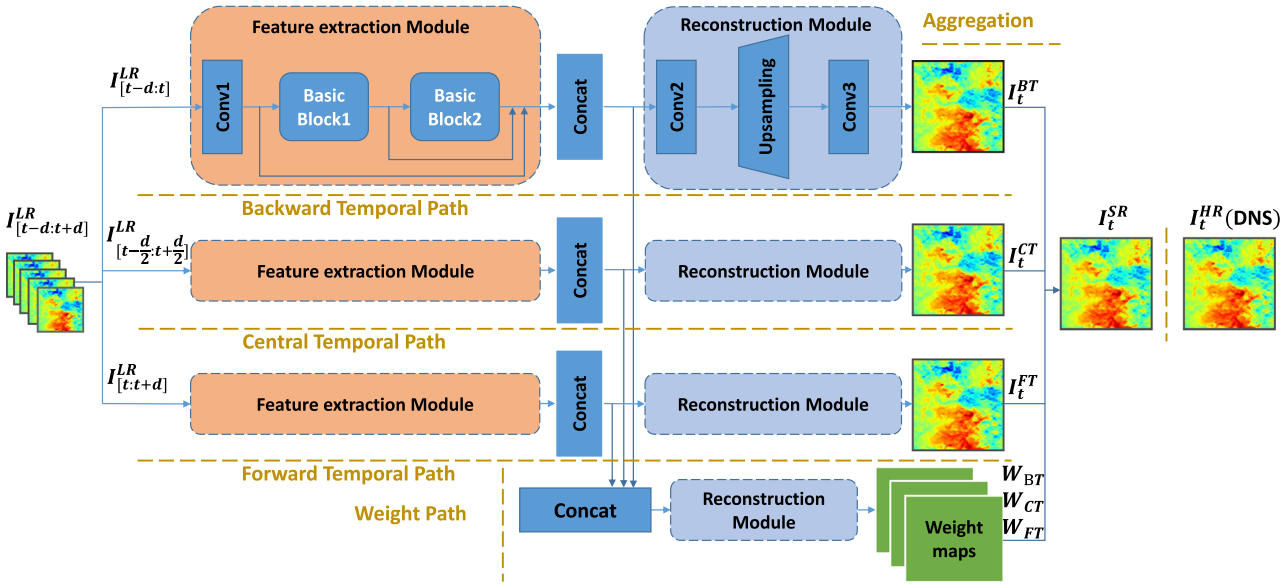


**FIG. 3**. Schematic diagram of the multiple temporal paths convolutional neural network (MTPC). It consists of three temporal paths and a weight path. The three temporal paths generate HR predictions $I^{BT}_t$, $I^{CT}_t$, and $I^{FT}_t$ based on the series of LR snapshots $I^{LR}_{[t-d:t]}$, $I^{LR}_{[t-\frac{d}{2}:t+\frac{d}{2}]}$, and $I^{LR}_{[t:t+d]}$, respectively. These predictions are combined based on the weight maps from the weight path to generate the final prediction $I^{SR}_t$.

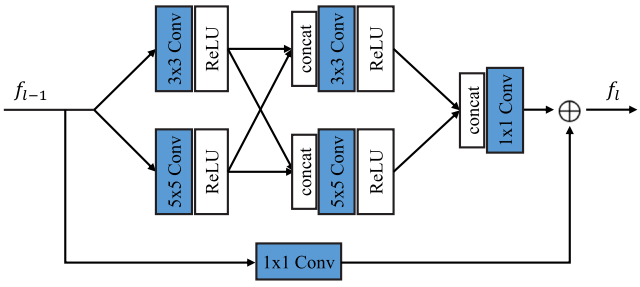


**FIG. 4**. The structure of the enhanced multi-scale residual block (EMSRB).

information without additional computational cost. The reconstruction module is made up of several convolutional layers and an upscaling operator. The upscaling operator used here is the PixelShuffle, which is also applied in the SCNN. The detailed parameters of the BTC are listed in Table II. These parameters are also applied to the other two paths for simplicity.

### 3. Weight path and aggregation

The weight path concatenates all the features extracted in temporal paths as the input of the reconstruction module. The reconstruction module in the weight path has an identical structure as that in temporal paths except for the last convolutional layer. In the last layer, the activation function is the sigmoid function, i.e., $\phi(x) = \frac{1}{1+e^{-x}}$, which normalizes data into the range of [0, 1]. The reconstruction module outputs the pixel-level weight maps with the shape of $rH \times rW \times C \times 3$. The final output $I_t^{SR}$ is obtained by the following operator:

$$I_t^{SR} = W_{BT} \odot I_t^{BT} + W_{CT} \odot I_t^{CT} + W_{FT} \odot I_t^{FT}, \tag{5}$$

where $\odot$ denotes pointwise multiplication. The loss function of the MTPC is

$$\mathscr{L}(I^{SR}, I^{HR}) = \mathscr{L}_2(I^{SR}, I^{HR}) + \lambda\Big(\mathscr{L}_2(I^{BT}, I^{HR}) + \mathscr{L}_2(I^{CT}, I^{HR}) + \mathscr{L}_2(I^{FT}, I^{HR})\Big), \tag{6}$$

**TABLE II**. Detailed parameters of the Backward Temporal Path (BTP) in the multiple temporal paths convolutional neural network (MTPC). Both basic block 1 and basic block 2 are the enhanced multi-scale residual block (EMSRB).

| Layer | Kernel size/filters | Activation function | Output size |
|---|---|---|---|
| Input | ... | ... | $H \times W \times C \times (d+1)$ |
| Conv1 | $3 \times 3/64$ | ... | $H \times W \times 64$ |
| Basic block 1 | ... | ReLU | $H \times W \times 64$ |
| Basic block 2 | ... | ReLU | $H \times W \times 64$ |
| Concat | ... | ... | $H \times W \times 192$ |
| Conv2 | $1 \times 1/C \times r^2$ | ... | $H \times W \times C \times r^2$ |
| Upsampling | ... | ... | $rH \times rW \times C$ |
| Conv3 | $3 \times 3/C$ | ... | $rH \times rW \times C$ |

where $\lambda$ is a constant to control the ratio between prediction results in branches and the final output. $\lambda = 0.5$ is chosen in this work.

### 4. Hyperparameter

The hyperparameter is a key factor that may significantly affect the performance of networks. Generally speaking, if the amount of data is sufficient, the performance of the networks would be enhanced with the increase in the number of the basic blocks, convolutional layers, filter maps, etc. Considering the balance of efficiency and accuracy, we only use two basic blocks in the feature extraction module. In addition, all the convolutional layers have 64 filter maps.

## III. DATASET AND IMPLEMENT DETAILS

Johns Hopkins turbulence databases[34,35] (JHTDB) are open-source simulation databases containing space-time histories of direct numerical simulations of several classical turbulent flow problems. Two datasets, namely, forced isotropic turbulence and turbulent channel flow,[36] from the JHTDB are used in this study.

All the DL models are trained and tested on the 2D slices of three-dimensional (3D) fluid fields. The velocity vector $\mathbf{u} = (u, v, w)$ is selected as the input quantity, so the channels $C = 3$. Here, we set $d = 2$, i.e., the MTPC takes $2d + 1 = 5$ consecutive frames as inputs. To be more efficient in the training process, we randomly sample $128 \times 128$ patches at each training iteration. Therefore, the shape of input LR frames to the SCNN and MTPC is $batchsize \times 128 \times 128 \times 3$ and $batchsize \times 128 \times 128 \times 15$, respectively. The Adam optimizer[37] is adopted to update the weights of both models.

## IV. RESULT

### A. Forced isotropic turbulence

To evaluate the DL models' ability for recovering the turbulent flows, the classical forced isotropic turbulence is selected as our first test. Many important statistical properties would be assessed here to check the performance at different spatial scales. In addition to the two DL models, the bicubic interpolation method is also examined for comparison. The isotropic turbulence data of the JHTDB are generated from a direct numerical simulation on a $1024^3$ mesh size at Taylor-scale Reynolds number $Re_\lambda = 418$. The size of the computational domain is $2\pi \times 2\pi \times 2\pi$ (the range of $x$, $y$, $z$ is $[0, 2\pi]$). The flow is governed by the incompressible Navier–Stokes equations,

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \nu \nabla^2 \mathbf{u}, \tag{7}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{8}$$

where $\mathbf{u}$ is the velocity vector, $p$ is the pressure, and $\nu$ is the kinematic viscosity. About five large-eddy turnover times are stored in the dataset. The training set is sampled in the first turnover time at $z = 0$ (x-y plane), and another 200 snapshots are selected as the validation set. Besides, 200 snapshots at $z = \pi/2$ in the last turnover time are chosen for testing. The input data are normalized by the maximum absolute value computed over the current frame. The $L_2$ error norms that are defined as $\varepsilon = \|I^{HR} - F(I^{LR}, \theta)\|_2 / \|I^{HR}\|_2$ are utilized

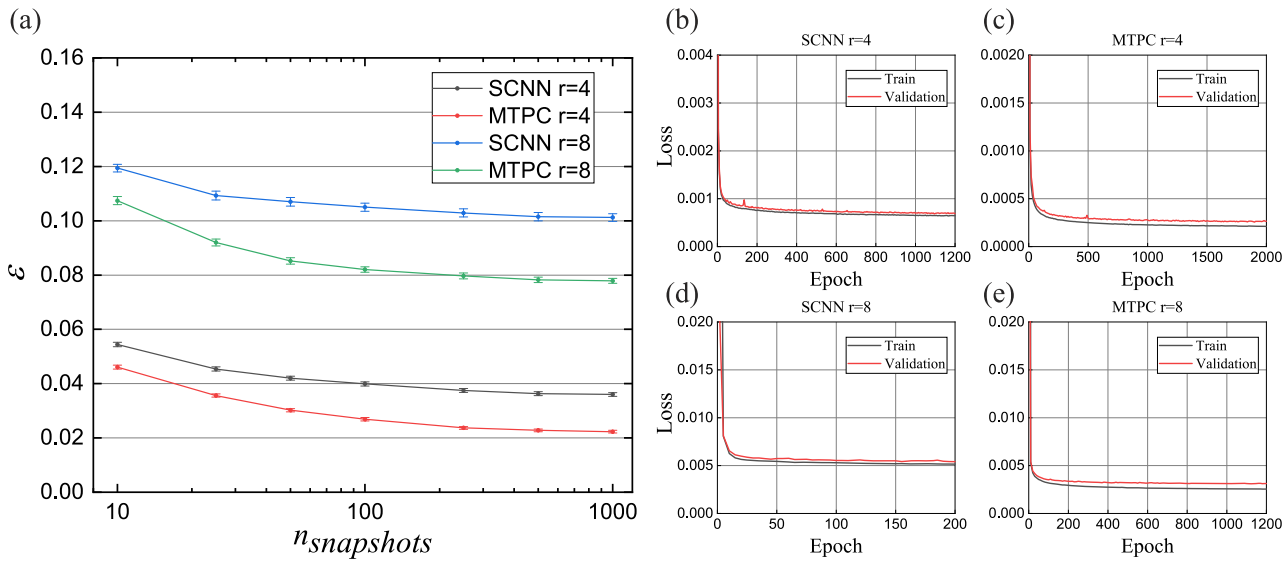


**FIG. 5**. (a) $L_2$ error norms on the validation set for the models trained with different data amounts. Learning curves for the SCNN and MTPC at [(b) and (c)] $r = 4$ and [(d) and (e)] $r = 8$ when the number of the snapshots for training $n_{snapshots}$ is 1000.

here as the measurement of error between the reconstructed flow field and that from the DNS. Two upscaling factors $r = 4$ and $r = 8$ are studied.

First, the influence of the number of training samples on the performance is checked. The $L_2$ error norms as a function of training data amounts are shown in Fig. 5(a). It is shown that $\varepsilon$ on the validation set decreases with the increase in $n_{snapshots}$ for both the SCNN and MTPC models. It seems that when the amount of data is 500, the error norms approach a plateau. The models' accuracy cannot be enhanced noticeably from increasing training data anymore. In other words, 1000 snapshots are enough to get a good result and after that the performances of the models become data-independent. Both models are trained with learning rate $lr = 10^{-4}$ and *batchsize* = 32. To avoid overfitting, the training procedure is stopped when the loss on the validation set no longer decreases significantly. The following results are obtained from models trained with 1000 snapshots, and the corresponding learning curves are shown in Figs. 5(b)–5(e).

Second, the performances of the models on the testing set are investigated. The $L_2$ error norms for the models are listed in Table III. It is seen that the error norms of the SCNN and MTPC are much smaller than those of the bicubic interpolation. Furthermore, among the three models, the MTPC achieves the minimum error and deviation at both scales, e.g., the average error norm of the MTPC at $r = 4$, is only 0.0283 with 0.0004 standard deviation. Therefore, the MTPC is the most accurate and stable model.

**TABLE III**. $L_2$ error norms and standard deviations of the reconstructed flow fields from bicubic interpolation, SCNN, and MTPC in the case of forced isotropic turbulence. The boldface values indicate the best result obtained in each case.

| | Bicubic | SCNN | MTPC |
|---|---|---|---|
| r = 4 | 0.1458 ± 0.0015 | 0.0449 ± 0.0006 | **0.0283** ± 0.0004 |
| r = 8 | 0.2575 ± 0.0022 | 0.1218 ± 0.0010 | **0.0985** ± 0.0007 |

Next, to demonstrate the capability of the DL models, we would like to perform a direct comparison. 64 × 64 patches from an instantaneous snapshot in each case were cropped, and the comparison is presented in Fig. 6. It seems that the patch generated by the bicubic interpolation is excessively smooth, e.g., the flow fields of $v$ in Figs. 6(a) and 6(b). Maybe it only captures the low-frequency information. The results of the SCNN and MTPC look closer to the DNS result than those of the bicubic interpolation, e.g., the flow fields of $u$ and $w$ in Fig. 6(b). A possible explanation is that the SCNN and MTPC enrich the low-resolution fluid field with high-frequency contents. Furthermore, the flow fields of $u$ and $w$ in Fig. 6(b) also clearly show that the MTPC supplements more details and reconstructs finer structures than the SCNN. It may be due to the additional temporal information that the MTPC includes. These arguments are also supported by the following analysis of the kinetic energy spectra.

In this section, several turbulent statistics are evaluated to check whether the reconstructed turbulent flow is physically reasonable. It is noticed that the measurement of the numerical error, e.g., the $L_2$ error norm, is not enough. To assess the performance of models at the inertial scale, the energy spectra are shown in Fig. 7(a). It is shown that all the proposed methods are able to reproduce the spectra at low wavenumbers. However, the bicubic interpolation, SCNN, and MTPC result at $r = 8$ gradually deviates from the DNS result at $k \approx 25$, $k \approx 50$, and $k \approx 60$, respectively. The SCNN result almost covers the range where $-5/3$ power law holds ($k < 60$), while the MTPC completely recovers the range. For the $r = 4$ case, the MTPC predicts almost all the inertial scales although some deviations exist at the scales where $E(k) < 10^{-5}$. Meanwhile, it is also seen that there is a challenge for the DL models to predict

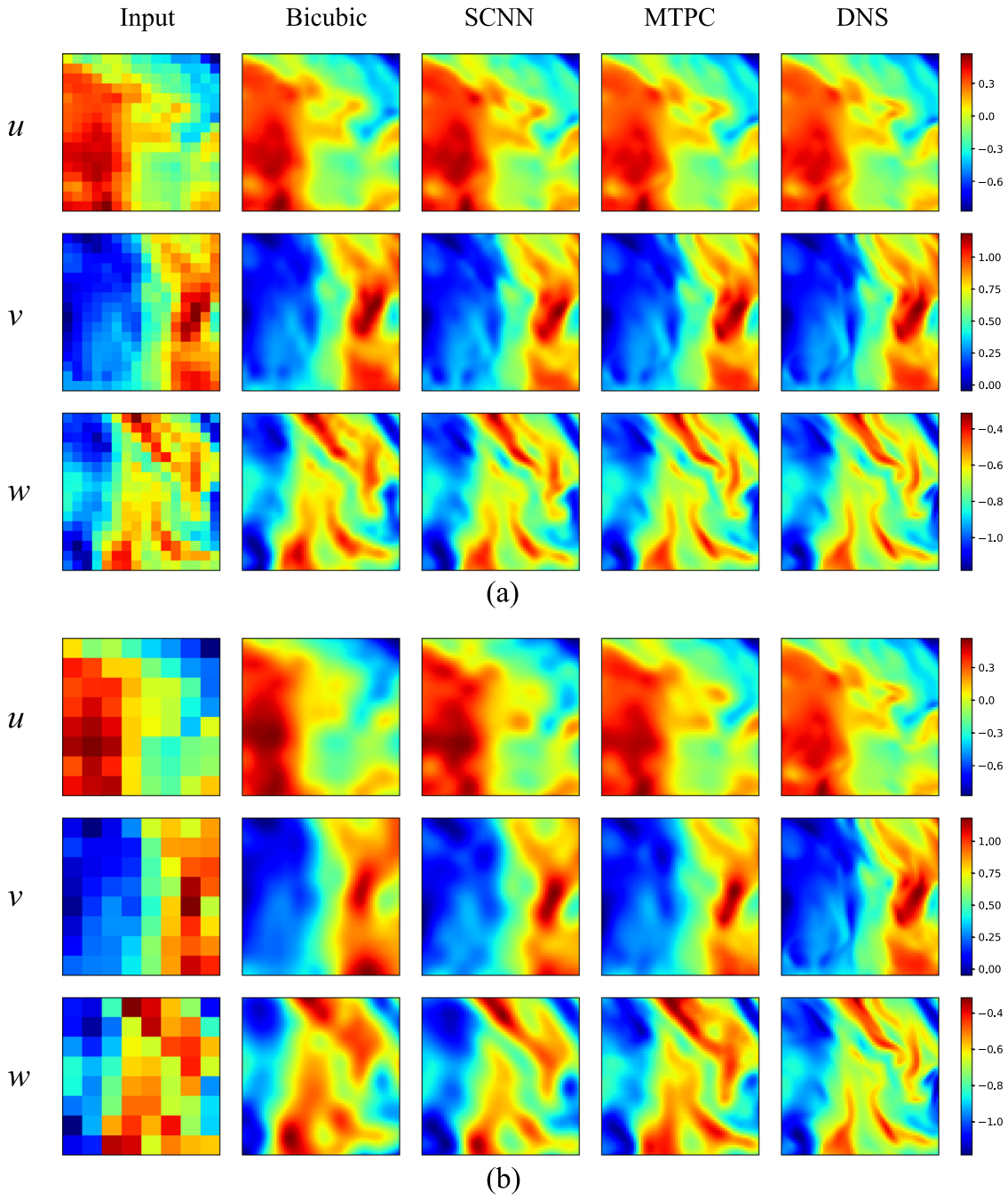


**FIG. 6**. Comparison of the low resolution, bicubic interpolation, SCNN, MTPC, and DNS results of the isotropic turbulence at (a) $r = 4$ and (b) $r = 8$. The inputs are scaled to the same size as the others for comparison.

the kinetic energy several orders of magnitude smaller than the total energy.

We would like to further assess the intermittency of velocity gradients to test the performance of models at the smallest scale. The probability density function (PDF) of the normalized velocity gradients for the reconstructed field and DNS data is shown in Fig. 7(b). It is seen that all the DL models captured the non-Gaussian feature, but the SCNN underestimates the PDF at tails. It reveals that the

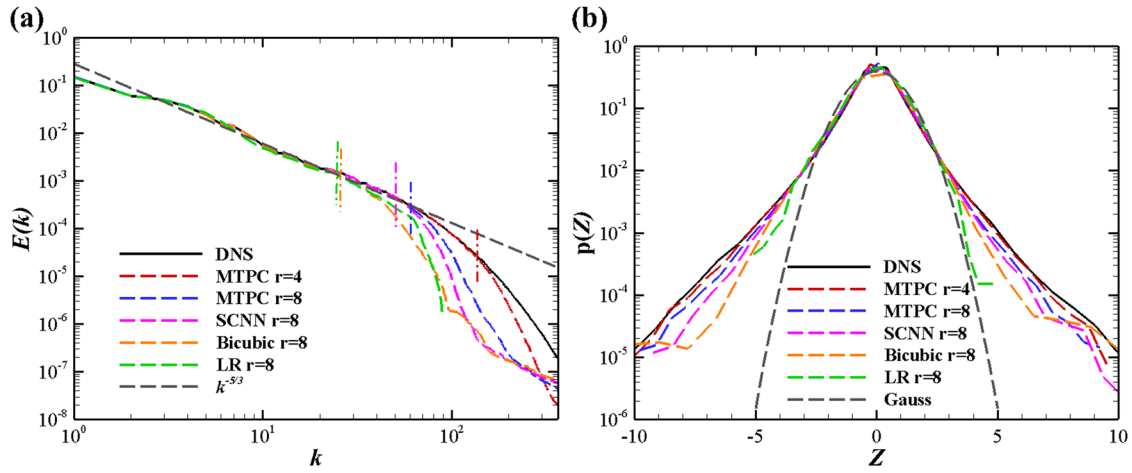


**FIG. 7**. (a) Kinetic energy spectra and (b) probability density functions of the normalized velocity gradients $p(Z)$ of the reconstructed isotropic turbulent flow fields and corresponding DNS data. Here, $k$ is the wavenumber, and $Z = \frac{\partial u'}{\partial x}$ is the partial derivative of velocity fluctuation $u'$ in the $x-$direction. The dashed–dotted lines in (a) denote the locations that the reconstructed results begin to deviate from the DNS result.

SCNN prediction may be less intermittent than the DNS result. On the other hand, the prediction of the MTPC is more consistent with the DNS result.

Finally, to check if the relation between different directions and velocity components is well captured, an additional test of the Q–R joint PDF of the velocity gradient tensor (VGT) for the MTPC is performed, where $Q$ and $R$ are the second and third invariants of VGT, respectively. The Q–R joint PDFs at $r = 4$ and $r = 8$ for the reconstructed field and DNS data are shown in Figs. 8(a) and 8(b), respectively. It is shown that the PDF at $r = 4$ is much closer to the DNS result than that at $r = 8$. The isolines of probability in the Q–R plane are perfectly predicted when $r = 4$. Hence, a smaller upscaling factor is preferred in the MTPC model. On the other hand, the classic tear-drop shape is still obtained when $r = 8$, i.e., the flow field is still well reconstructed.

In summary, the above performances show that the DL models can reproduce isotropic turbulence with high fidelity. In terms of all the above criteria, the performance of the MTPC is better than that of the SCNN and bicubic interpolation method.

## B. Channel flow

The above forced isotropic turbulence may be the simplest turbulence. Most turbulent flows are anisotropic and may be bounded by solid surfaces in reality. In this section, we would like to investigate the performances of these models on a more complex turbulent flow, i.e., a turbulent channel flow. Using the DL models to reconstruct the turbulent channel velocity field is a tougher task due to the anisotropic property, the three components of the velocity maybe not in the same order of magnitude.

The channel flow database used here also comes from JHTDB. It was obtained from a DNS with periodic boundary conditions in the longitudinal ($x$) and transverse ($z$) directions, a no-slip boundary condition was applied at the top and bottom walls ($y/h = \pm 1$, $h$ is

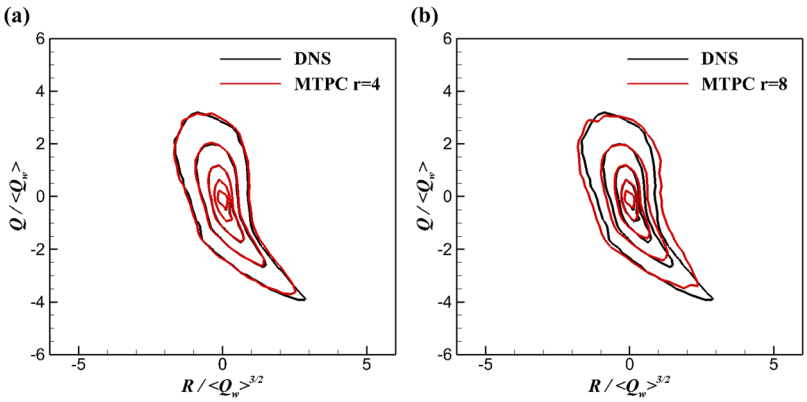


**FIG. 8**. Q–R joint PDF of the reconstructed isotropic turbulent fields and corresponding DNS data at (a) $r = 4$ and (b) $r = 8$. Q and R are the second and third invariants of the velocity gradient tensor, respectively.

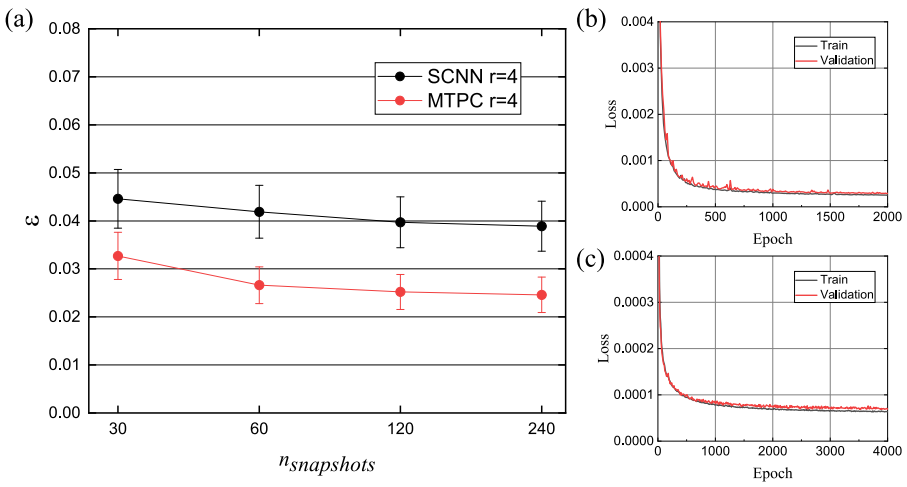


**FIG. 9**. (a) $L_2$ error norms on the validation set for the models trained with various amounts of data. The learning curves of (b) the SCNN and (c) the MTPC when the number of snapshots for training $n_{snapshots}$ is 120.

the half-channel height). To solve the incompressible Navier–Stokes equations, a wall-normal, velocity–vorticity formulation[38] was used. The simulation was performed at friction velocity Reynolds number $Re_\tau = 9.9935 \times 10^2$ on a $2048 \times 512 \times 1536$ ($N_x \times N_y \times N_z$) grid for about a single flow-through time. The domain length is $L_x \times L_y \times L_z = 8\pi h \times 2h \times 3\pi h$. More details about the simulation can be found in the work of Graham *et al.*[36]

In this section, only the snapshots on the wall-parallel $x$–$z$ plane are considered. In the wall-bounded turbulent flow, according to the distance to the wall $y$, the boundary layer can be divided into two layers, i.e., the outer layer ($0.2\delta \le y \le \delta$) and the inner layer ($y \le 0.2\delta$), where $\delta$ is the thickness of the boundary layer. Usually, the normalized distance to the wall is defined as $y^+ \equiv \frac{u_\tau y}{\nu}$, where $u_\tau$ is the friction velocity and $\nu$ is the kinematic viscosity. The inner layer can be further divided into three regions, i.e., viscous sublayer ($y^+ \le 5$), buffer layer ($5 \le y^+ \le 30$), and log-law region ($30 \le y^+ \le 0.2\delta$). Each region has its own specific property. For the simplicity, in the following description, the viscous sublayer, buffer layer, log-law region, and outer layer are denoted by regions I, II, III, and IV, respectively.

To test our SCNN and MTPC models, wall-parallel snapshots are chosen from regions I, II, III, and IV at a ratio of 1 : 1 : 2 : 2 because less data are available inside regions I and II in the original database. Totally 30 snapshots are sampled as validation data. Because the orders of magnitude of three velocity components ($u$, $v$, and $w$) in the channel flow may be different, the maximum absolute value of each velocity component in the current frame is adopted to normalize the corresponding velocity component separately. Both models are trained with learning rate $lr = 10^{-4}$ and $batchsize = 16$. The validation error as a function of data amounts is shown in Fig. 9(a). It is shown that the data amount $n_{snapshots} = 120$ is enough for the SCNN and MTPC to get good results. Hence, the following results are all obtained from models trained with 120 snapshots. The corresponding learning curves are shown in Figs. 9(b) and 9(c).

The testing set consists of 30 snapshots at the other $y^+$ locations, which are not included in the training set. It is also noted that the testing data are sampled 10 000 DNS time steps away from the training data. In this way, we can check whether the DL models are

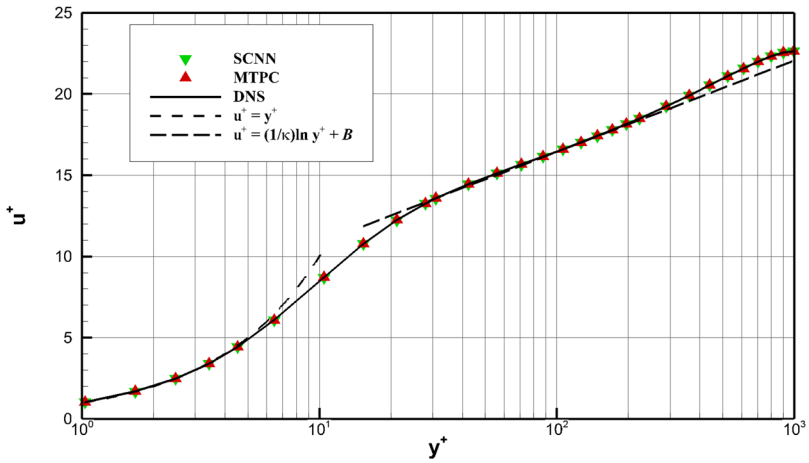


**FIG. 10**. Mean velocity profiles for the predicted results and corresponding DNS result in the case of channel flow. $\kappa$ is von Karman constant, and $B$ is a constant. Here, $\kappa = 0.41$, $B = 5.2$.

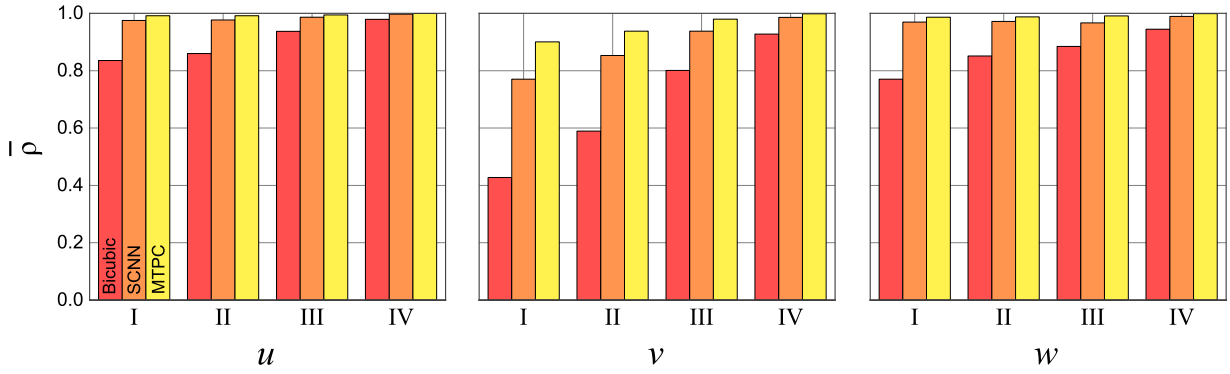


**FIG. 11**. The average correlation coefficients between the reconstructed channel flow fields and corresponding DNS results. The red, orange, and yellow columns represent the performances of the bicubic interpolation, SCNN, and MTPC models, respectively.

able to grasp various statistical properties in different flow regions. Here, only $r = 4$ case is investigated.

First, the mean velocity profiles are checked. The predicted mean velocity profiles are shown in Fig. 10. It is shown that both the SCNN and MTPC predict the $u^+$ profile well and captures the linear property in the viscous sublayer as well as the $\frac{1}{\kappa}$ logarithmic law property in the log-law region, where the von Karmen constant $\kappa = 0.41$ here.

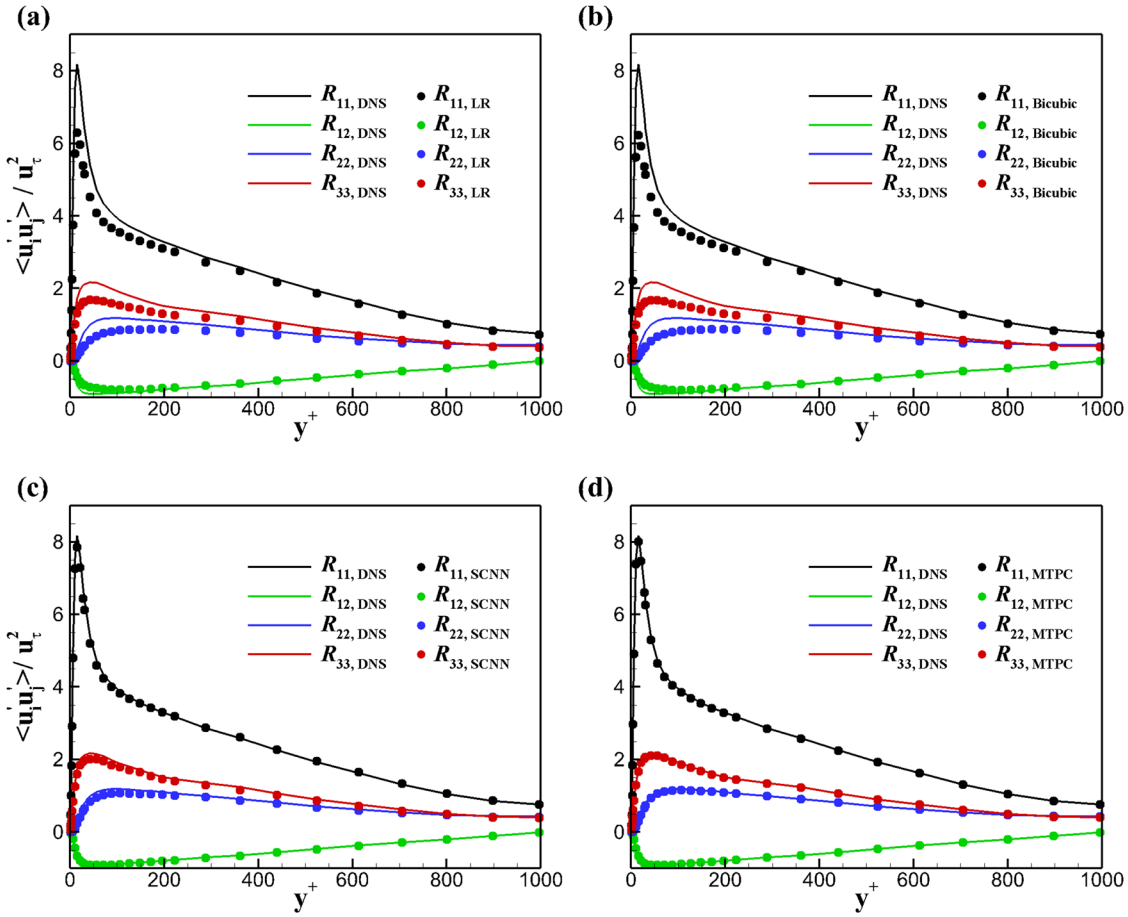


**FIG. 12**. Reynolds stresses normalized by the friction velocity against $y^+$ from DNS of channel flow and (a) low resolution input, (b) bicubic interpolation, (c) SCNN, and (d) MTPC result. Here, $R_{ij} = \langle u_i' u_j' \rangle / u_\tau^2$ and $u_i'$ is the velocity fluctuation ($i = 1, 2, 3$ for streamwise, wall-normal, and spanwise directions, respectively).

Second, the performances of the DL models on prediction of $u$, $v$ and $w$ in regions I to IV are evaluated. The correlation coefficients $\rho(I^{SR}, I^{HR})$ are used to quantify the difference between the prediction and the reference, i.e.,

$$\rho(I^{SR}, I^{HR}) = \frac{\sum_i (I_i^{SR} - \langle I^{SR}\rangle)(I_i^{HR} - \langle I^{HR}\rangle)}{\sqrt{\sum_i (I_i^{SR} - \langle I^{SR}\rangle)^2}\sqrt{\sum_i (I_i^{HR} - \langle I^{HR}\rangle)^2}}, \quad (9)$$

where $\langle\cdot\rangle$ denotes the ensemble average. The larger the coefficient $\rho$ is, the more accurate the prediction is.

The correlation coefficients between the reconstructed data and the DNS data are shown in Fig. 11. It is shown that for the velocity components $u$ and $w$, all three reconstruction methods achieve high $\rho$. However, for $v$, the correlation coefficient is relatively low. Therefore, it seems more difficult to predict the non-mainstream velocity component. It is also shown that for each model, the prediction in region IV is better than those in regions I and II, especially for $v$. This may be attributed to the fact that the turbulent property is abundant and changes intensively inside the regions very close to the wall.

We also observe that among each group of columns, which consist of red, orange, and yellow columns, the yellow one is always the highest. Besides, the heights of all yellow columns are very close to unity, and even the shortest yellow column has a height of about 0.9 (see regions I for $v$). Hence, correlations between the MTPC predictions and the DNS results are very high. The performance of the MTPC model seems always the best among the three models.

Third, the performances of the DL models on prediction of Reynolds stresses are investigated. The Reynolds stresses that are relevant to velocity fluctuation as a function of $y^+$ are shown in Fig. 12. Figure 12(b) looks like Fig. 12(a). It seems that the bicubic interpolation does not improve the LR input and almost underestimates all components of Reynolds stresses seriously. Predictions obtained from the SCNN [see Fig. 12(c)] are in reasonable agreement with the DNS data, except for small deviations at the peaks. On the other hand, the MTPC well reproduced all components of Reynolds stresses [Fig. 12(d)].

Finally, an additional test to check the predicted spatial structures is performed. An intuitive demonstration of the reconstructed flow fields at the outer layer is shown in Fig. 13. It is seen that compared with the bicubic and SCNN cases, in terms of the contours of $u$ and $w$, the MTPC predicts more small scales in the contours, which are very close to those in the DNS result. Hence, the performance of the MTPC model is better than those of the bicubic and SCNN models. The kinetic energy spectra at the location $y^+ = 10.45$ are also analyzed. The result is shown in Fig. 14. It is seen that energy spectra produced from the low-resolution input are not able to follow the trend of the DNS result even at low wavenumbers. However, the MTPC result, taking LR data as an input, agrees well with the DNS result in the most range. It seems that the MTPC does not take the low-frequency information in the input as the whole truth but amends it and then supplements the high-frequency contents. For the MTPC model, the kinetic energy spectrum of $v$ seems not so good as that of $u$ and $w$. The possible reason is that the kinetic energy in the $y$ direction is two orders of magnitude smaller than that in the $x$ direction. From Fig. 14, it is also shown that all the kinetic energy spectra generated from the MTPC drop faster than those of the DNS at high wavenumbers. Hence, it is still a challenge to reconstruct small scale spatial structures.

All the experiments are conducted on the NVIDIA RTX 2080 graphics processing unit (GPU). The training processes of the SCNN

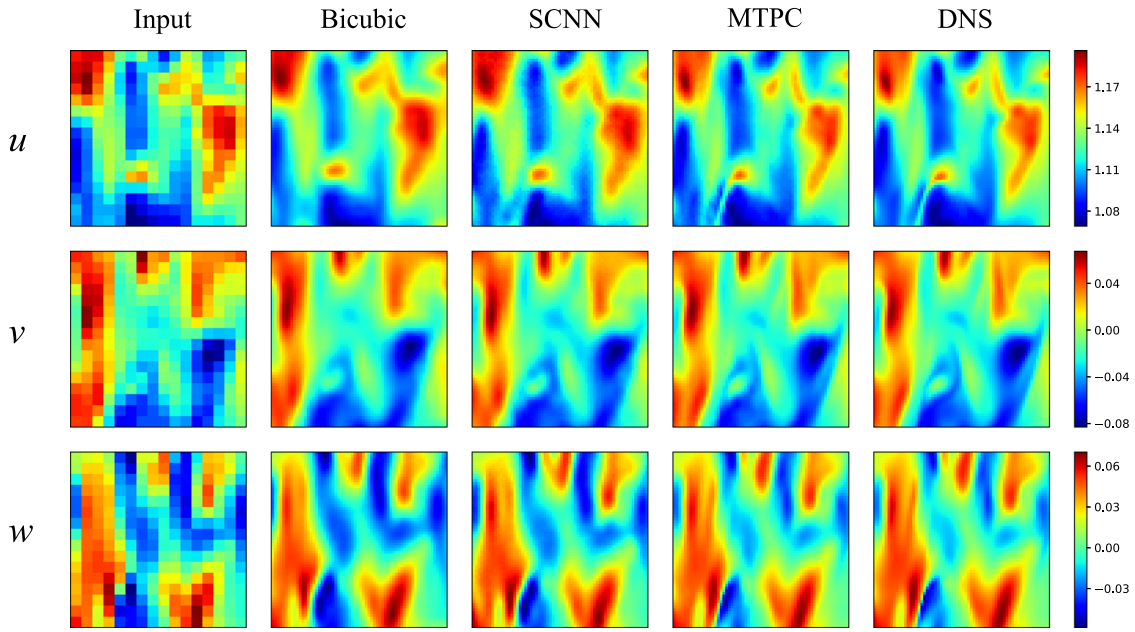


**FIG. 13**. Comparison of the low resolution, bicubic interpolations, SCNN, MTPC, and DNS results of turbulent channel flow at $r = 4$. The low-resolution inputs are scaled to the same size as the other results for comparison.

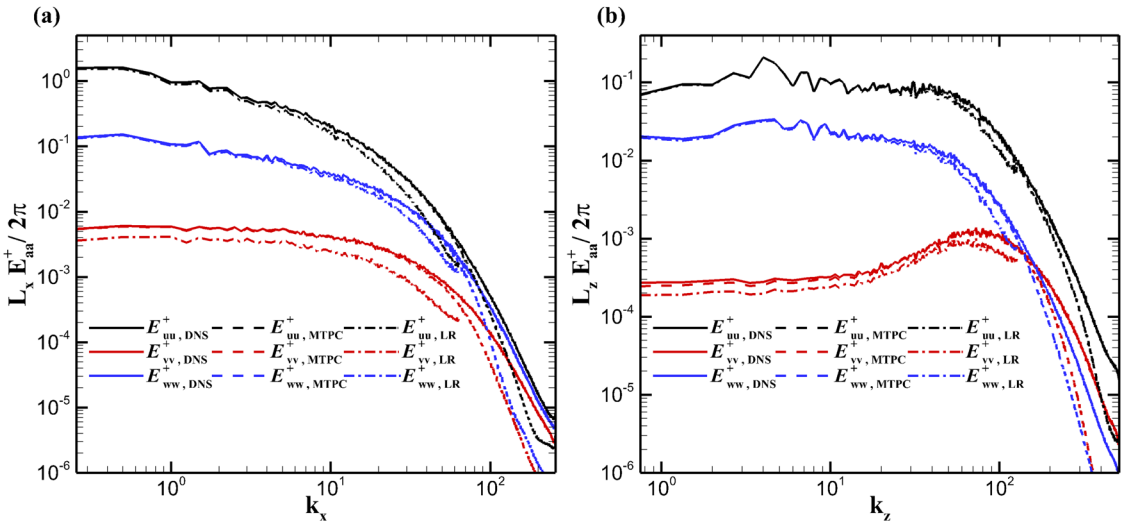


**FIG. 14**. (a) Streamwise power spectra and (b) spanwise power spectra at $y^+ = 10.45$ as functions of $k_x$ and $k_z$ in the case of turbulent channel flow.

and MTPC for the channel flow take about 2.5 h (4.5s/*epoch* × 2000) and 21.9 h (19.7s/*epoch* × 4000), respectively. On the other hand, reconstructing a shape of the 1536 × 2048 × 3 flow field from 384 × 512 × 3 LR flow data only takes approximately 0.03 s and 0.38 s for the SCNN and MTPC models, respectively.

## V. CONCLUSIONS

Two deep learning models, i.e., the SCNN and MTPC, are developed for the super-resolution reconstruction of turbulent flows. They all take low-resolution flow information as an input. However, the SCNN takes one instantaneous snapshot as an input, while the MTPC takes a temporal sequence of snapshots as an input. Therefore, the MTPC has the advantage of drawing extra temporal information from adjacent frames.

To see whether the deep learning methods are able to reproduce turbulence, two canonical turbulent problems were tested. For the isotropic turbulence, the reproduced energy spectra and predicted PDF of the normalized velocity gradients by the MTPC are close to those of the DNS result. For the turbulent channel flow, the deep-learning-based approaches greatly enhanced the spatial resolution in different wall regions and layers. The correlation coefficients between reconstructed data and the reference data are high in the outer layer and log-law region where turbulence dominates. However, the coefficients are not so high in the viscosity-dominated region where there are the most vigorous turbulent activities. All assessments in both cases show that the MTPC greatly improved the quality of the LR input and outperformed the SCNN and bicubic interpolation method, especially at small scales. The extra temporal information from consecutive snapshots helps the MTPC to generate more physically reasonable results. On the other hand, because there are more input snapshots to handle, the MTPC spends more time on prediction compared with static models. All the experiments were performed with 2D snapshots, but the networks used in this study can be easily extended to 3D cases by using 3D convolution kernels.

Although deep-learning methods achieve great progress, there are still some challenges. First, the DL models are not able to exactly reproduce the kinetic energy several orders of magnitude smaller than the total energy, i.e., the very small spatial structures. Second, the DL methods show performance diversity in different directions in the anisotropic turbulence case even though a special normalization method is used to convert data in the three directions into the same order of magnitude. These problems may be eliminated by introducing novel machine learning methods like unsupervised learning or including prior physical knowledge in the training process. That may be part of our future work.

The success of the DL models in reconstructing subgrid flow variables probably inspires the development of subgrid models in the CFD. The SR technology can also serve as the post-processing tool to denoise, correct, or enrich data from experiments and numerical simulations. We believe that the super-resolution technology would have a board practical application in the fluid dynamics with the increasing volumes of data accessible.

## ACKNOWLEDGMENTS

This work was supported by the Natural Science Foundation of China (NSFC) (Grant Nos. 11872064, 11572312, and 11621202).

## REFERENCES

[1] R. J. Adrian, "Twenty years of particle image velocimetry," Exp. Fluids **39**, 159–169 (2005).
[2] S. L. Brunton and B. R. Noack, "Closed-loop turbulence control: Progress and challenges," Appl. Mech. Rev. **67**, 050801 (2015).
[3] J. N. Kutz, "Deep learning in fluid dynamics," J. Fluid Mech. **814**, 1–4 (2017).
[4] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems

involving nonlinear partial differential equations," J. Comput. Phys. **378**, 686–707 (2019).
[5]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).
[6]V. Sekar, Q. Jiang, C. Shu, and B. C. Khoo, "Fast flow field prediction over airfoils using deep learning approach," Phys. Fluids **31**, 057103 (2019).
[7]Z. Wang, K. Luo, D. Li, J. Tan, and J. Fan, "Investigations of data-driven closure for subgrid-scale stress in large-eddy simulation," Phys. Fluids **30**, 125101 (2018).
[8]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).
[9]B. D. Tracey, K. Duraisamy, and J. J. Alonso, "A machine learning strategy to assist turbulence model development," in *53rd AIAA Aerospace Sciences Meeting* (AIAA, 2015), p. 1287.
[10]X. Jin, P. Cheng, W.-L. 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).
[11]H. Koizumi, S. Tsutsumi, and E. Shima, "Feedback control of Karman vortex shedding from a cylinder using deep reinforcement learning," AIAA Paper No. 2018-3691, 2018.
[12]J. Rabault and A. Kuhnle, "Accelerating deep reinforcement learning strategies of flow control through a multi-environment approach," Phys. Fluids **31**, 094105 (2019).
[13]M. Gazzola, A. A. Tchieu, D. Alexeev, A. de Brauer, and P. Koumoutsakos, "Learning to school in the presence of hydrodynamic interactions," J. Fluid Mech. **789**, 726–749 (2016).
[14]S. Verma, G. Novati, and P. Koumoutsakos, "Efficient collective swimming by harnessing vortices through deep reinforcement learning," Proc. Natl. Acad. Sci. U. S. A. **115**, 5849–5854 (2018).
[15]K. Hayat, "Multimedia super-resolution via deep learning: A survey," Digital Signal Process. **81**, 198–217 (2018).
[16]S. L. Brunton, B. R. Noack, and P. Koumoutsakos, "Machine learning for fluid mechanics," Annu. Rev. Fluid Mech. **52**, 477–508 (2020).
[17]C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *European Conference on Computer Vision* (Springer, 2014), pp. 184–199.
[18]W. Dong, L. Zhang, G. Shi, and X. Wu, "Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization," IEEE Trans. Image Process. **20**, 1838–1857 (2011).
[19]J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," IEEE Trans. Image Process. **19**, 2861–2873 (2010).
[20]C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *European Conference on Computer Vision* (Springer, 2016), pp. 391–407.
[21]W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep Laplacian pyramid networks for fast and accurate super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2017), pp. 624–632.
[22]M. Haris, G. Shakhnarovich, and N. Ukita, "Recurrent back-projection network for video super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2019), pp. 3897–3906.
[23]K. Fukami, K. Fukagata, and K. Taira, "Super-resolution reconstruction of turbulent flows with machine learning," J. Fluid Mech. **870**, 106–120 (2019).
[24]H. Gunes and U. Rist, "Spatial resolution enhancement/smoothing of stereo–particle-image-velocimetry data using proper-orthogonal-decomposition–based and kriging interpolation methods," Phys. Fluids **19**, 064101 (2007).
[25]S. Cai, S. Zhou, C. Xu, and Q. Gao, "Dense motion estimation of particle images via a convolutional neural network," Exp. Fluids **60**, 73 (2019).
[26]Y. Lee, H. Yang, and Z. Yin, "PIV-DCNN: Cascaded deep convolutional neural networks for particle image velocimetry," Exp. Fluids **58**, 171 (2017).
[27]Z. Deng, C. He, Y. Liu, and K. C. Kim, "Super-resolution reconstruction of turbulent velocity fields using a generative adversarial network-based artificial intelligence framework," Phys. Fluids **31**, 125111 (2019).
[28]A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems* (MIT Press, 2012), pp. 1097–1105.
[29]S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems* (MIT Press, 2015), pp. 91–99.
[30]O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention* (Springer, 2015), pp. 234–241.
[31]W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2016), pp. 1874–1883.
[32]S. B. Pope, *Turbulent Flows* (Cambridge University Press, 2000).
[33]J. Li, F. Fang, K. Mei, and G. Zhang, "Multi-scale residual network for image super-resolution," in *Proceedings of the European Conference on Computer Vision (ECCV)* (Springer, 2018), pp. 517–532.
[34]Y. Li, E. Perlman, M. Wan, Y. Yang, C. Meneveau, R. Burns, S. Chen, A. Szalay, and G. Eyink, "A public turbulence database cluster and applications to study Lagrangian evolution of velocity increments in turbulence," J. Turbul. **9**, N31 (2008).
[35]E. Perlman, R. Burns, Y. Li, and C. Meneveau, "Data exploration of turbulence simulations using a database cluster," in *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing* (ACM, 2007), p. 23.
[36]J. Graham, K. Kanov, X. Yang, M. Lee, N. Malaya, C. Lalescu, R. Burns, G. Eyink, A. Szalay, R. Moser *et al.*, "A web services accessible database of turbulent channel flow and its use for testing a new integral wall model for LES," J. Turbul. **17**, 181–215 (2016).
[37]D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv:1412.6980 (2014).
[38]J. Kim, P. Moin, and R. Moser, "Turbulence statistics in fully developed channel flow at low Reynolds number," J. Fluid Mech. **177**, 133–166 (1987).