# Artificial Neural Network Modeling Small-Scale Turbulence of Isotropic Turbulent Flows

Jiangtao Tan[1,2] and Guodong Jin[1,2(✉)]

[1] The State Key Laboratory of Nonlinear Mechanics, Institute of Mechanics, Chinese Academy of Sciences, Beijing 100190, China
tanjiangtao17@mails.ucas.ac.cn, gdjin@lnm.imech.ac.cn

[2] School of Engineering Sciences, University of Chinese Academy of Sciences, Beijing 100049, China

**Abstract.** Small-scale fluid motions play an important role in the relative dispersion and clustering of inertial particles in turbulent flows. In this paper, an artificial neural network (ANN) is used to recover the small-scale turbulence of isotropic turbulent flows in the context of *a prior* large-eddy simulation (LES). The nonlinear convection and pressure gradient terms in the governing equations of subgrid-scale (SGS) velocity are treated as output labels $C_i$, and the large-scale velocity and the velocity gradient tensor are taken as input features. The data required for training and testing of the ANN are provided by the Direct Numerical Simulation (DNS) and filtered Direct Numerical Simulation (FDNS). The optimized model highly fits the relationship between input features and output labels. Using the ANN model and large-scale flow field information, the approximate governing equation of small-scale motions is solved numerically, and the small-scale flow field can be obtained. The developed flow field can be statistically consistent with the results of DNS. The probability density function (PDF) of small-scale velocity and velocity gradient tensor are consistent with those of DNS. Our research indicates that the small-scale flow field can be calculated by combining the large-scale flow field with the ANN methods. Furthermore, we combine the ANN model with FDNS to predict the statistics of heavy particles as *a prior* LES. The results show that the *a prior* LES with ANN model can improve the prediction accuracy in clustering and relative velocity of particle pairs. This study provides a feasible method for constructing small-scale turbulence, which can reduce the amount of computation compared with DNS and used to the study of small-scale turbulent mixing.

**Keywords:** Large-eddy Simulation · Small-scale model · Artificial Neural Network · Inertial particles · Turbulent clustering

## 1 Introduction

Large eddy simulation (LES) is an important method for the numerical study of turbulence. Its basic principle is to solve only the filtered Navier-Stokes equations containing large-scale motion and establish a sub-grid scale (SGS) model to account for the influence of small-scale motion on large-scale motion and to close filtered N-S equations


X. Zheng and S. Balachandar (Eds.): IUTAM 2023, IUTAM Bookseries 41, pp. 40–57, 2024.
https://doi.org/10.1007/978-3-031-47258-9_3

[1, 2]. Small-scale turbulence is closely related to the energy dissipation of turbulence, which has the best prospect of being universal or quasi-universal [3]. The intermittency of small-scale turbulence affects the propagation of sound waves in the atmosphere and can strengthen the mixing effect on substances in combustion. In turbulent noise and turbulent two-phase flow, small-scale motion plays an important role in the relative dispersion and non-uniform preferential concentration of particles, turbulent noise high-frequency signals and the deformation of droplets or bubbles [4]. Deepening the study of small-scale turbulence can improve the accuracy of LES of flow field simulation at high Reynolds numbers.

In LES, the large scales of motion containing energy are numerically simulated, while unsolved sub-grid scales and their interactions with large scales are modeled [5]. On the one hand, SGS stresses are modeled to close the filtered N-S equations in LES. The classic SGS models include the Smagorinsky model [6], the scale similarity model [7, 8], the gradient model [9], the mixed model [10], the dynamic model [11–17], etc. The study of SGS stress has always been a hot topic in LES. Germano et al. [12] proposed a method to calculate the variation of the coefficients of the SGS eddy viscosity model with space and time. The mathematical inconsistency in the formula of this method limits the practicability of the model. Ghosal et al. [18] corrected these inconsistencies, and the new model can be used for the general non-uniform flow. On the other hand, the study of SGS velocity is also an important aspect of LES. However, the model established by the coupling of approximate deconvolution and synthetic turbulence and the stochastic differential equation method still has some shortcomings in simulating the intermittent characteristics of small-scale motion [19, 20]. It is still a difficult problem to build a high-fidelity SGS model.

The breakthrough in computer technology has led to a comprehensive improvement in computing capabilities. Therefore, algorithms such as machine learning and deep learning have played an important role in various research fields. Machine learning is a learning method that simulates humans through big data, and deep learning aims to extract intrinsic features from data. These methods have made breakthroughs in image classification, natural language processing, and face recognition. MacCulloch and Pitts [21] first proposed the concept of machine learning in their computational research on the principles of biological neural networks published in 1943. Later, with the famous "Turing Test", machine learning, an artificial intelligence research method, began to enter the field of scientific research. The modeling of machine learning mainly includes data processing, feature selection, building a framework, and optimizing parameters. Researchers in the computer field continuously propose and optimize different algorithms for different processes. For example, according to learning methods, it can be divided into supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. The supervised learning data itself has tags, and the existing tags are used as expected results to optimize the model. The supervised learning algorithm is usually divided into classification algorithms and regression algorithms, including linear models, artificial neural networks, support vector machines, and so on. Unsupervised learning data itself does not carry labels, and machines explore the inherent relationships between the data. For example, clustering algorithms classify samples without categories based on the similarity between the samples. In semi-supervised

learning, only part of the data is labeled, resulting in low learning costs and high accuracy. Reinforcement learning (RL) is an area of machine learning inspired by behaviorist psychology, concerned with how software agents ought to take actions in an environment to maximize some notion of cumulative reward. The most familiar example is the AlphaGo that blooms in the Go game. Machine learning methods are widely used, and have achieved breakthrough results in image classification, natural language processing, and face recognition. Face recognition, intelligent capture, etc. in daily life have the use of machine learning algorithms behind them. The interactive processing method of parallel computing for big data has improved the adaptability of machine learning in big data environments. Data sampling and attribute selection also provide modeling ideas for machine learning to process big data issues [22]. Deep learning allows a computing model composed of multiple processing layers to learn data with multiple levels of abstraction. These methods have greatly improved the technical level of many other fields such as speech recognition, visual object recognition, object detection, and drug discovery and genomics. Deep convolutional neural networks have made breakthroughs in processing images, video, speech, and audio, while recursive networks have brought research prospects for sequential data such as text and speech [23].

Artificial neural network (ANN) is a popular and widely used method in machine learning (ML), which was first proposed in the 1950s. ANN combines neuroscience and connects the theoretical model of biological neural networks - the McCulloch-Pitts [21] (M-P) neuron model - into a network to simulate the operation of neurons in the human brain. ANN utilizes the computing power of computers to process and analyze data and has excellent data processing and modeling capabilities. However, in the past, limited by the performance of computers, neural network methods once fell into a slump. After entering the 21st century, ANN has once again demonstrated its ability to fit data with deep learning. Deep learning, in a narrow sense, can be considered a multi-layer neural network. Data is crucial for deep learning, as too little data can easily lead to model overfitting. In the era of big data, deep learning has become more closely related to various disciplines and research fields. The ability to learn in-depth has not been disappointing and has played an extremely important role in various studies.

The application of machine learning to turbulence modeling has been increasingly discussed. The universal approximation theorem [24] states that any function may be approximated by a sufficiently large and deep neural network. Brunton et al. [25] discussed that ML is a useful technique for deriving information from data. He explained the classification of ML and analyzed several successful cases of modeling through ML. There are many opportunities and challenges in applying ML to the field of fluid dynamics. Yarlanki et al. [26] used ML algorithms to optimize the CFD model, resulting in a 35% reduction in model error compared to the k-ε model. Wang et al. [27] used random forest methods for turbulence modeling. The model was trained by Direct Numerical Simulation (DNS) data. The Reynolds stress in different flows was predicted, and the study has achieved good predictive effects. Fukami et al. [28] performed super-resolution of low-resolution turbulent flow field data with the convolutional neural network to reconstruct an accurate turbulence model, which promoted the efficient super-resolution model of fluid flow. Qu et al. [29] extracted turbulence features through convolutional neural networks, which reduced the error compared with previous work.

Gamahara and Hattori [30] used ANN to study SGS stress models, showing that ANN is a promising tool for building new SGS models. Sarghini et al. [31] used a multilayer feedforward ANN in LES to build an SGS model, with the flow field velocity as the input and the turbulence viscosity coefficient as the output. The model was optimized by the error backpropagation algorithm. Their results showed that the computation time is shortened by 20% compared with the existing models without considering the time of neural network training, which proves the feasibility of using ANN to study turbulence dynamics. Xie et al. [32] used spatial artificial neural network (SANN) modeling to improve the prediction accuracy of SGS stress and SGS heat flow in the compressible isotropic turbulence model, which has a higher correlation and smaller error than the gradient model. The SANN model has better prediction results for speed, temperature, and instantaneous flow structure than the dynamic hybrid model in a posteriori comparison. In the study of isotropic turbulence, Zhou et al. [33] obtained training data by a spectral method, constructed a single hidden layer feed-forward ANN, took the velocity gradient tensor and filter size as input, SGS stress as output, predicted the energy transfer rate under different Reynolds numbers and filter sizes. The correlation of the training results of the ANN model was higher than 0.9, which was significantly improved compared with the traditional model. Beck et al. [34] rewrote the N-S equations for viscous compressible fluids as: $\partial U/\partial t + R(FU) = 0$. The ANN and deep learning models have been established for the $R(FU)$ terms and speed of the three dimensions. Their results showed that a model with higher stability and accuracy was obtained. Milano and Koumoutsakos [35] used an ANN to simulate and predict near-wall turbulent flow fields. Yin et al. [36] used an ANN to establish a ML model for turbulence feature quantity prediction, and the results showed that when the Reynolds stress error is small, the error of predicting the average flow is also small. Xie et al. [37] compared the modeling effect of three different ANN models on SGS stress in LES, and the correlation coefficient of SGS stress of the ANN model was improved in the prior analysis compared with the velocity gradient model, among which the relative error of the spatial ANN model and the deconvolutional artificial neural networks model was less than 15%. In a posterior verification, the prediction accuracy of the turbulence field of the three models was higher.

Based on the theory of LES, the current study modifies the governing equations of small-scale flows. The nonlinear term of small-scale velocity and pressure gradient in the equation are modeled using ANN to achieve an approximate linearization of the governing equation. The sum of the nonlinear term of small-scale velocity and pressure gradient is defined as model parameters. By numerically solving the linearized equation, velocity information of small-scale flows can be obtained. The ANN model constructed in the study uses large-scale velocities and velocity gradient tensor as inputs, and model parameters as outputs, and is optimized using an error backpropagation algorithm. The data required for training and testing of the ANN are obtained from DNS and filtered Direct Numerical Simulation (FDNS) of homogeneous isotropic turbulence. The prediction accuracy and generalization of the trained ANN model are evaluated using DNS data at various Reynolds numbers and filter widths.

The paper is organized as follows: The linearization of small-scale flow governing equations and the derivation of model parameters are introduced in Sect. 2. In Sect. 3, the

theory of neural networks, error backpropagation algorithms, and network optimization schemes are introduced. In Sect. 4, the optimization results of neural networks are introduced, including the use of hyperparameters and error loss functions. In Sect. 5, we substitute the ANN model into a linearized governing equation to numerically solve small-scale flows and verify the accuracy of the model results through DNS data and FDNS data. Specifically, we tested the probability density distribution curve of velocity and velocity gradient, velocity gradient tensor joint probability density distribution, energy spectrum, etc. Section 6 gives the conclusions of this paper.

## 2 Governing Equations

This paper takes fully developed homogeneous isotropic turbulence as the modeling object. Isotropic turbulence is a hypothetical turbulence model, but its isotropic properties exist in the fine structures of anisotropic turbulence [38]. This study uses neural network modeling methods to study the characteristics of isotropic turbulence, which can expand the modeling ideas of isotropic turbulence and lay the foundation for the study of anisotropic turbulence. The data used in the study were derived from databases for direct numerical simulation of uniform isotropic turbulence and filtered direct numerical simulation.

The pseudo-spectral method is used to process the N-S equations and uses Fourier transform to transform the turbulent flow field from the physical field to the spectral space. After solving the N-S equations in the spectral space, the inverse Fourier transform is used to convert the velocities in spectral space back to the physical space. The periodic boundary conditions of isotropic turbulence make Fourier expansion accurate and effective [39]. FDNS uses the turbulence data obtained from DNS to filter out Fourier modes greater than the cutoff wave number in the turbulence energy spectrum through filtering processing and then inverts them back into physical space to obtain a large-scale velocity field. The small-scale velocity field is obtained by subtracting the large-scale velocity field from the full-scale velocity field [20, 40].

The modeling parameters of the ANN model are derived from the Navier-Stokes equations. The Navier-Stokes equations for incompressible flows are:

$$\frac{\partial u_i}{\partial t} + \frac{\partial u_i u_j}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 u_i}{\partial x_j \partial x_j}, \tag{1}$$

$$\frac{\partial u_i}{\partial x_i} = 0. \tag{2}$$

In the study of LES, the governing equation of large-scale flow is obtained by filtering the flow field.

$$\frac{\partial \overline{u}_i}{\partial t} + \frac{\partial \overline{u_i u_j}}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \nu \frac{\partial^2 \overline{u}_i}{\partial x_j \partial x_j}. \tag{3}$$

From Eq. (1) and Eq. (3), the governing equation for the filtered small-scale flow can be obtained.

$$\frac{\partial u_i'}{\partial t} + \frac{\partial \left(u_i u_j - \overline{u_i u_j}\right)}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p'}{\partial x_i} + \nu \frac{\partial^2 u_i'}{\partial x_j \partial x_j}, \tag{4}$$

where $u_i$ denotes the full-scale velocity, $p$ is the pressure, $\rho$ is the fluid density, $\nu$ is the kinematic viscosity, $\overline{u}_i$ is the large-scale velocity, and $u_i'$ is the small-scale velocity component. The nonlinear unclosed terms in Eq. (4): $\left(u_i u_j - \overline{u_i u_j}\right)$ is processed as follows:

$$u_i u_j - \overline{u_i u_j} = u_i u_j - \overline{u}_i \overline{u}_j - (\overline{u_i u_j} - \overline{u}_i \overline{u}_j) = u_i u_j - \overline{u}_i \overline{u}_j - \tau_{ij} = (\overline{u}_i + u_i')(\overline{u}_i + u_j') - \overline{u}_i \overline{u}_j - \tau_{ij} = \overline{u}_i u_j' + u_i' \overline{u}_j + u_i' u_j' - \tau_{ij}, \tag{5}$$

where $\tau_{ij} = (\overline{u_i u_j} - \overline{u}_i \overline{u}_j)$ donates the sub-grid scale (SGS) stress. Combine Eq. (5) with Eq. (4) to obtain Eq. (6) and Eq. (7):

$$\frac{\partial u_i'}{\partial t} + \frac{\partial (\overline{u}_i u_j' + u_i' \overline{u}_j + u_i' u_j' - \tau_{ij})}{\partial x_j} = -\frac{1}{\rho} \frac{\partial p'}{\partial x_i} + \nu \frac{\partial^2 u_i'}{\partial x_j \partial x_j'}, \tag{6}$$

$$\frac{\partial u_i'}{\partial t} + \overline{u}_j \frac{\partial u_i'}{\partial x_j} + u_j' \frac{\partial \overline{u}_i}{\partial x_j} - \nu \frac{\partial^2 u_i'}{\partial x_j \partial x_j} - \frac{\partial \tau_{ij}}{\partial x_j} = C_i. \tag{7}$$

On the left side of Eq. (7) are the linear terms for small-scale velocity $u_i'$ and large-scale velocity $\overline{u}_i$, which represent the acceleration, the convection of small-scale by the large-scale velocity $\overline{u}_i$, the effects of the shear deformation on the small scales, viscous dissipation, and sub-grid stress, respectively. The right side of the Eq. (7) represents the sum of the nonlinear convection and pressure gradient terms,

$$C_i = -\frac{1}{\rho} \frac{\partial p\prime}{\partial x_i} - \frac{\partial u_i' u_j'}{\partial x_j}, \tag{8}$$

which shall be modeled. Due to the interaction between large-scale flow and small-scale flow in the flow field, it is believed that large-scale velocity and velocity gradient influence $C_i$. Therefore, we used the ANN model to establish a mapping between large-scale velocity, velocity gradient tensor, and model parameter $C_i$.

## 3 ANN Modeling

In biological neural networks, neuronal cells use the specialized nature of the outer membrane to receive signals from other neurons through dendrites, integrate these signals by the cell body, and then transmit information to other neurons through axons to transmit information. Artificial Neural Network is a machine learning method that simulates biological neural networks in computer science research. Artificial neural networks mimic biological neurons, using a neuron-like M-P model (see Fig. 1) [41] as the basic unit to build the model.

In the ANN, $x_j$ represents the input features, $y_i$ represents the output labels, $\omega_{ij}$ denotes the weight of each input, $\theta_i$ specifies the corresponding threshold, $f$ is the activation function (refers to the function of mapping the neuron inputs to the outputs, generally a step function or sigmoid function) to process the input values. The expression for output $y_i$ is written as:

$$y_i = \sum\nolimits_{j=1}^{n} f(\omega_{ij} x_j - \theta_i). \tag{9}$$

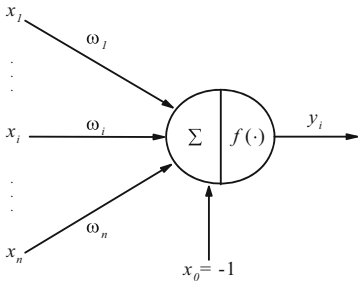


**Fig. 1.** M-P model of a neural cell.

Different neurons are connected by the outputs of the previous layer as the inputs of the next layer, forming an overall model of the network of neurons. In this study, the main use of feedforward neural networks - neurons are arranged in layers, and the neurons in each layer only connect with neurons in adjacent layers before and after, and neurons between the same layer and the compartment do not make connections (see Fig. 2).

In this study, the model was improved by changing the number of neuronal layers and the number of neurons, and the weight $\omega_{ij}$ and threshold $\theta_i$ in the neural network were optimized by the error Back Propagation algorithm [42]. We use the error back-propagation algorithm to optimize the model. First, we calculate the root mean square error $E_k$ between the ANN output $y$ and the theoretical value $y_0$.

$$E_k = \frac{1}{n}\sum\nolimits_{j=1}^{n}\left(y_j - y_{j0}\right)^2. \tag{10}$$

Then calculate the negative gradient of the error $E_k$ with respect to each of the weights $\omega_{ij}$, thresholds $\theta_i$ on the error representing their effect on the final output y.

$$\Delta\omega_{ij} = -\eta(\partial E_k/\partial\omega_{ij}), \tag{11}$$

$$\Delta\theta_i = -\eta(\partial E_k/\partial\theta_i), \tag{12}$$

where $\eta$ donates the learning rate of the model, which controls the amplitude of each parameter update. A too large learning rate will cause the optimization parameters to fluctuate near the optimal value, and a too small learning rate can cause the model to be optimized too slowly. The update expression for the parameter is:

$$\omega_{ij} \leftarrow \omega_{ij} + \Delta\omega_{ij}, \tag{13}$$

$$\theta_i \leftarrow \theta_i + \Delta\theta_i. \tag{14}$$

Finally, through the continuous iteration of this method, the error between the output value of the model and the theoretical value can be gradually reduced. When the error is less than a given small number, the model training is complete.

In the study, we used a total of 12 physical quantities, including 3 components of large-scale velocity $\overline{\boldsymbol{u}}$ and 9 components of velocity gradient tensor $\nabla\overline{\boldsymbol{u}}$, as inputs to

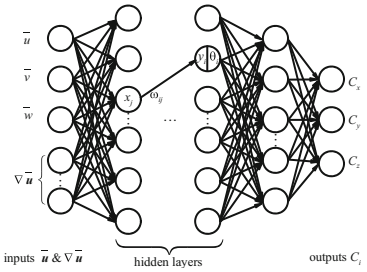


**Fig. 2.** Feedforward Neural Network. The inputs include large scale velocities $\overline{u}, \overline{v}, \overline{w}$ and velocity gradient tensor $\nabla\overline{\boldsymbol{u}}$. Output $C_i$ contains components in the $x$, $y$ and $z$ directions.

the neural network. With the model parameter $C_i$ as outputs, a fully connected neural network with 5 hidden layers and 128 network nodes per hidden layer is constructed. The flow fields with Taylor Reynolds numbers 64.43 and 128.78 were trained. Since the input has multiple features, to ensure that each feature has the same impact on the network during initial training, the maximum and minimum values of each feature are used to normalize the data.

$$x = \frac{x - x_{\min}}{x_{max} - x_{min}} \tag{15}$$

The normalized data is in the [0, 1] range, and can be restored through inverse normalization. To ensure the applicability of the model, the numerical data is randomly divided into training data sets ($X\_train$, $y\_train$) and test data sets ($X\_test$, $y\_test$). The amount of data in the training data set is larger than that in the test data set. The training dataset is brought into the neural network model to optimize the weights and threshold, and the test dataset is substituted into the optimized model to calculate the loss function size to verify the accuracy of the model.

The sigmoid function is used instead of the step function to avoid discontinuity in the substitution of the step function itself. The properties of the two are roughly similar, mainly used to indicate that they are activated when the input quantity is multiplied by the weight and the threshold value is greater than 0. The output is 1, otherwise, the output is 0.

The optimization of the network takes the loss function as a reference. This model is essentially a regression problem that uses mean square error as the loss function. The expression is written as:

$$cost = \frac{1}{n}\sum_{j=1}^{n}\left(y_{pre} - y_0\right)^2 \tag{16}$$

The model randomly divides the data into training data sets and test data sets according to a ratio of 9:1 between the training set and the test set. The model adopts a small batch training method. Small batch training refers to taking sample values from a training dataset. Because the data exists in a unified sample, it is considered that there should be the same inherent laws in each data. Therefore, the amount of data selected should not affect the training results of the final model. On this basis, small batch training is to

subdivide each round of training into $n_{batch}$ training sessions, with each epoch randomly capturing batch_size pieces of data in the training dataset. The relationship between $n_{batch}$, batch_size, and the total amount of data in the training dataset training_data_size, is:

$$n_{batch} = \frac{training\_data\_size}{batch\_size} \tag{17}$$

## 4 Training and Validation

Model optimization is a process of continuously adjusting parameters and iterative training. Hyperparameters refer to parameters that have been set before the model starts training. The hyperparameters in this model mainly include the number of neurons in the hidden layer, initial learning rate, decay-rate, momentum size, small batch training rounds, and small batch data size. Different hyperparameters have different effects on the fitting degree and operation speed of the network. The study adopted a group of excellent super parametric combinations: the learning rate was 0.001, and the decay-rate was 0.96, which means that the learning rate decayed to 0.96 times the initial value per 1000 steps. The validation of the cost function with the training steps is shown in Fig. 3.

Initially, the weight parameter inside the model is set randomly, and the threshold parameter is 0. Therefore, the initial error is very large. With the training of the ANN model, the error backpropagation algorithm continuously updates the weights and biases, and the error decreases rapidly. The learning process is well-formulated since the training and validation losses are nearly stable after 500 global iterations. The errors in the training and validation datasets exhibit similar variations, which shows that the ANN model has been successfully trained.

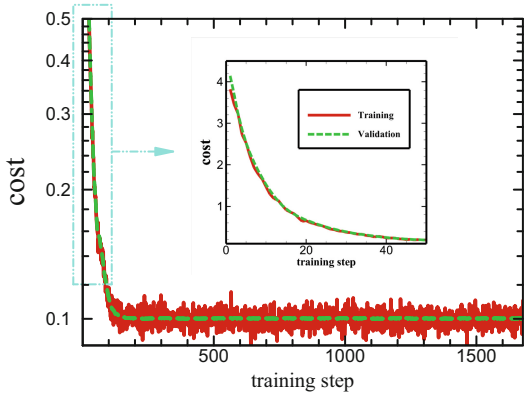


**Fig. 3.** Training and validation cost with the training step.

## 5 Results and Discussion

After training the ANN model, we can calculate the model parameter $C_i$ by inputting the large-scale velocity and velocity gradient tensor of the initial flow field. Then, by substituting $C_i$ into the governing equation and combining the initial value of the small-scale velocity, we can obtain the subsequent small-scale flow field numerically. Subsequent calculations only need the large-scale flow velocity from FDNS. The small-scale velocity is derived from the initial flow field, the large-scale velocity, and the governing equations.

The calculation process is shown in Fig. 4. First, input the initial flow field information - including large-scale flow and small-scale flow velocity. Then, calculate the model parameter $C_i$ of the current flow field through the ANN model, and substitute it into the governing equation to numerically solve the small-scale flow field velocity at the next time-step. The large-scale flow at the later time-step is calculated by FDNS. In this way, the flow field information at the next time-step can be obtained. By iterating this process, the velocity of the flow field at any time can be calculated.

In this paper, we have deduced the flow field from the initial time to 500 time-steps, approximately more than 30 Kolmogorov timescales. The flow field at this time was used for data analysis to verify the effectiveness of the ANN model. We used two sets of flow field data with Reynolds numbers of 64.43 and 128.78 to evaluate the ANN model. The data are derived from DNS, FDNS, and ANN model. We compare the data obtained by the ANN model with the data calculated by DNS and FDNS.

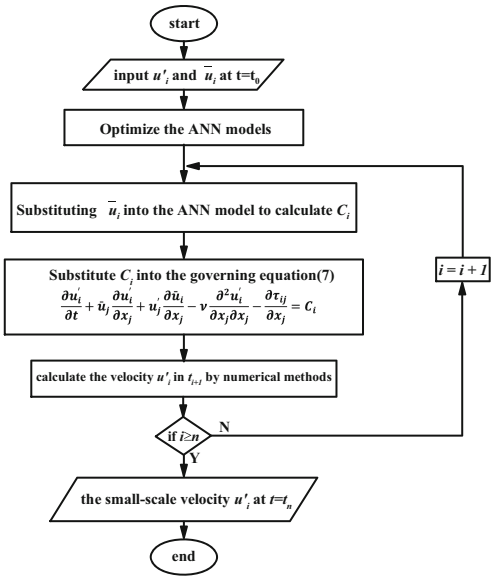


**Fig. 4.** Flow chart of small-scale velocity prediction.

Figure 5 shows the vortex contour for different models at the same time. In the area where the vortex structures on the right side change significantly in Fig. 5(a), the

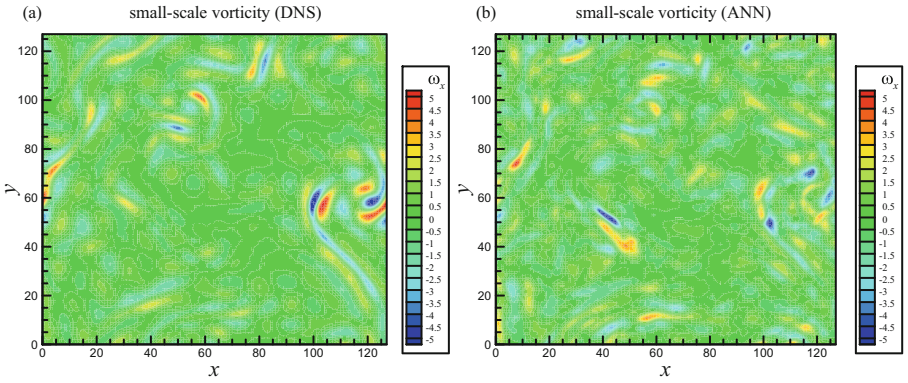


**Fig. 5.** Two-dimensional contours of small-scale vorticity ($Re_\tau = 64.43$). (a) the small-scale vorticity calculated by DNS and FDNS; (b) the small-scale vorticity calculated by ANN models.

structures obtained from the ANN model in Fig. 5(b) are consistent with those obtained by DNS. It can be seen from the contour that the ANN model can predict the occurrence of vortex well, and the vortex structures in the flow can be well reproduced. The flow field calculated by the ANN model can recover the structural characteristics of small-scale flow fields to a certain extent.

The probability density functions (PDFs) of small-scale velocity obtained by DNS and ANN model are shown in Fig. 6. Figure (a) shows the result at $Re_\tau = 64.43$, and Figure (b) shows the result at $Re_\tau = 128.78$. Unlike the Gaussian PDFs of the full-scale velocity, we can observe that the PDFs of small-scale velocity behaves apart from the Gaussian distribution at the tails. Furthermore, the ANN results of both sets of data can well match the DNS data. The results show that the ANN network is feasible in reconstructing the velocity distribution of small-scale flow fields.

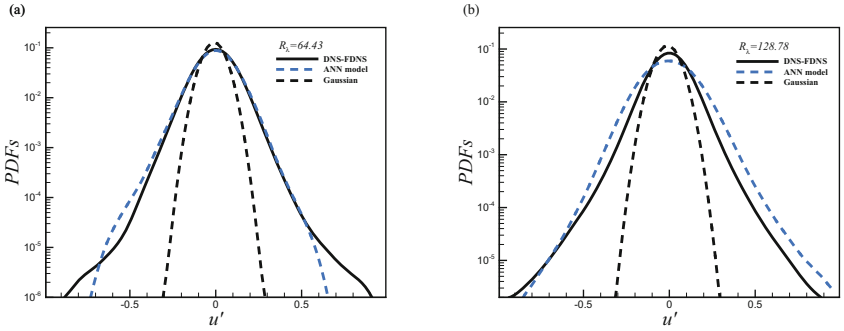


**Fig. 6.** The PDF of small-scale velocity. (a): Re = 64.43, (b): Re = 128.78.

Intermittency is an important property of small-scale turbulence. Statistically, intermittency is characterized by the non-Gaussian nature of the probability density functions of turbulent dissipation and velocity gradients. The non-Gaussian nature becomes

increasingly evident as the Reynolds number increases. On the PDF diagram, the tails will significantly deviate from the Gaussian curve, and the probability of data at both ends will significantly increase. Figure 7 is a comparison of the PDFs of the velocity gradients from the ANN model and DNS data, where Fig. 7 (a) shows the result at $Re_\tau = 64.43$, and Fig. 7 (b) shows the result at $Re_\tau = 128.78$. Statistically, the ANN model can recover the intermittency of the small structures in turbulent flows.

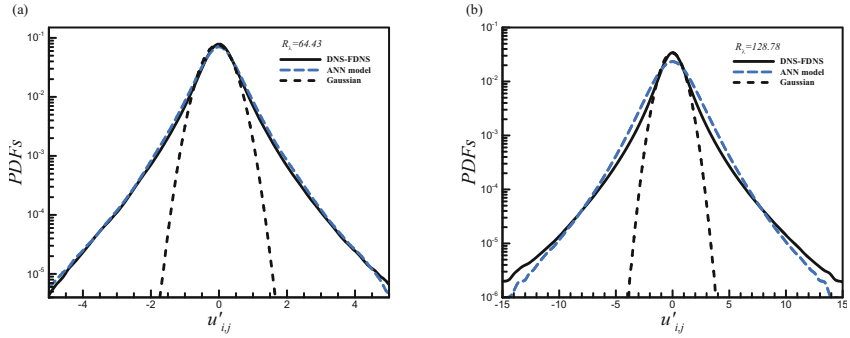


**Fig. 7.** The PDF of small-scale velocity gradient tensor. (a): Re = 64.43, (b): Re = 128.78.

To further examine the capability of the ANN model to recover the turbulent structures [43], the Q-R joint probability density distribution of the velocity gradient tensor was examined (see Fig. 8). Q and R are the second and third invariants of the velocity gradient tensor, respectively, $Q \equiv \frac{1}{2}\left(u_{i,i}^2 - u_{i,j}u_{j,i}\right)$ and $R \equiv Det(u_{i,j})$ [44]. According to the comparison of two sets of data, we can observe that the ANN model performs a good reconstruction of the flow structures.

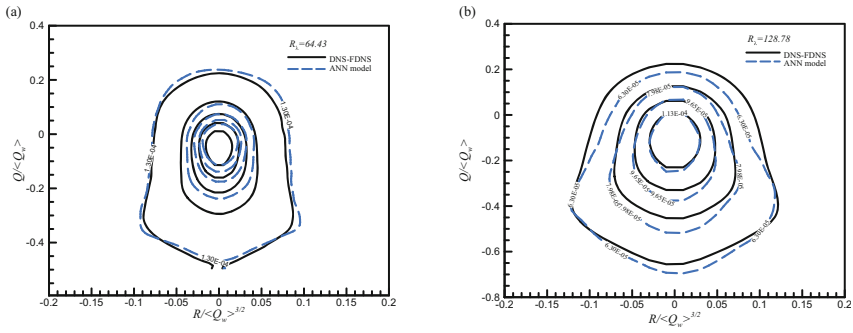


**Fig. 8.** Q–R joint PDF of the ANN model and corresponding DNS data at (a) Re = 64.43 and (b) Re = 128.78. Q and R are the second and third invariants of the velocity gradient tensor.

The energy spectrum of turbulent velocity is displayed in Fig. 9 for the DNS, FDNS, and ANN models. Due to the sharp-spectral filter used in FDNS, when the wave number $k$ is greater than the cutoff wavenumber, the energy spectrum value is 0. The flow field

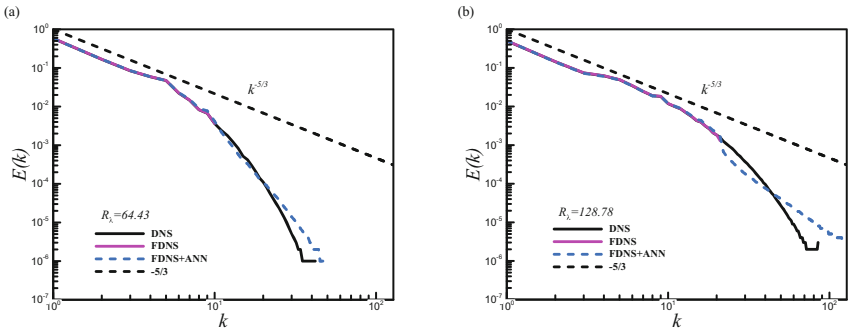


**Fig. 9.** Energy spectra of the isotropic turbulent flow fields obtained from DNS coupled with FDNS and the ANN model. (a) Re = 64.43, (b) Re = 128.78.

energy spectrum calculated by the ANN model can well fit the DNS curves when the wave number is smaller than the cutoff wavenumber. When the wave number is larger than the truncated wave number, the curve trend is consistent with that of DNS. According to the transformation between physical space and Fourier space, the larger the wave number, the smaller the corresponding physical spatial scale. Therefore, the ANN model can supplement small-scale information of the flow field, and the obtained energy spectrum is relatively accurate.

Next, we apply the ANN model to a prior LES of the statistics of inertial particles in isotropic turbulent flows. We calculate the motion of the inertial particles using DNS, FDNS and FDNS with the ANN model by tracking the Lagrangian trajectory of individual particles. The ANN model combined with FDNS is used to recover the missing small-scale turbulent motions in FDNS and drive the motion of particles.

The equations of motion of inertial particles are

$$\begin{cases} \frac{d\mathbf{x}_p(t)}{dt} = \mathbf{v}_p(t) \\ \frac{d\mathbf{v}_p(t)}{dt} = \frac{[\mathbf{u}^*(\mathbf{x}_p(t),t)-\mathbf{v}_p(t)]f}{\tau_p} \end{cases} \tag{18}$$

where $\mathbf{x}_p(t)$, $\mathbf{v}_p(t)$ are the particle position and velocity at time $t$, $\mathbf{u}^*\big(\mathbf{x}_p(t), t\big)$ is the fluid velocity seen by the particle, which can be calculated by a three-dimensional sixth-order Lagrangian interpolation scheme. $\mathbf{u}^*\big(\mathbf{x}_p(t), t\big)$ is used to represent $\mathbf{u}$ in DNS, $\tilde{\mathbf{u}}$ in FDNS and $\tilde{\mathbf{u}} + \mathbf{u}'_{ANN}$ in FDNS with the ANN model. $f$ denotes the nonlinear drag coefficient $f(Re_p) = 1 + 0.15Re_p^{0.687}$, $Re_p = \big|\mathbf{u}^* - \mathbf{v}_p\big|d_p/\nu$ is the particle Reynolds number, $\tau_p = \rho_p d_p^2/18\rho_f\nu$ is the particle response time, and $d_p = 0.4\eta$ and $\eta$ is the Kolmogorov length scale. To characterize the inertia of particles, we define the Stokes number

$$St = \frac{\tau_p}{\tau_\eta}, \tag{19}$$

where $\tau_\eta$ is the turbulent Kolmogorov timescale. We set St = 0.5, 1, and 2 in this study.

In order to validate the performance of the ANN model in a prior LES of particle relative motions, which is a long-lasting challenge for LES of small-scale process, we

statistically analyze the Radial Distribution Function (RDF) and the Radial Relative Velocity (RRV) of particle pairs [45–47]. The Radial Distribution Function is defined as the relative probability of finding a pair of particles at a given separation distance within a unit volume, compared to the expected number of pairs in a uniform distribution of particles:

$$g(r) = \frac{Q_{p,r}/\Delta V_r}{Q_p/V} = \frac{\frac{Q_{p,r}}{\frac{4}{3}\pi\left[\left(r+\frac{1}{2}dr\right)^3-\left(r-\frac{1}{2}dr\right)^3\right]}}{\frac{\frac{1}{2}N_p(N_p-1)}{(2\pi)^3}} \tag{20}$$

where $N_p$ is the total number of particles in the turbulent flow. The number of particle pairs is given by $Q_p = \frac{1}{2}N_p(N_p - 1)$. $Q_{p,r}$ represents the average number of particles found within the spherical shell volume $\Delta V_r$ with a distance of $r = |r|$ from the detecting particle. Here, $\Delta V_r = \frac{4}{3}\pi\left[\left(r + \frac{1}{2}dr\right)^3 - \left(r - \frac{1}{2}dr\right)^3\right]$, and $V = (2\pi)^3$ represents the total volume of the flow field. The RDF $g$(r) is a measure of particle preferential concentration, or clustering in turbulent flows. The Radial Relative Velocity of particle pairs is defined as:

$$w_r(r) = \left[\boldsymbol{v}_{p1}\left(\boldsymbol{x}_{p1}, t\right) - v_{p2}\left(\boldsymbol{x}_{p2}, t\right)\right] \cdot \frac{\boldsymbol{l}}{|\boldsymbol{l}|} \tag{21}$$

where $\boldsymbol{v}_{p1}\left(\boldsymbol{x}_{p1}, t\right)$ and $v_{p2}\left(\boldsymbol{x}_{p2}, t\right)$ are the velocities of two particles located at $\boldsymbol{x}_{p1}$ and $\boldsymbol{x}_{p2}$, respectively, and the relative separation between the two particles is $\boldsymbol{l} = \boldsymbol{x}_{p1} - \boldsymbol{x}_{p2}$. We test the relative separation between particles, $r \in [0.4\eta, 6.0\eta]$, divided into 280 segments. For each detecting particle pairs, we search particles within a spherical shell with a thickness of $0.02\eta$ to calculate the RDF and RRV.

Figure 10 (a) (c) (e) compares the curves of RDF as a function of dimensionless distance for St = 0.5, 1.0, and 2.0, in DNS, FDNS, and FDNS + ANN. Compared to DNS, FDNS exhibits significant differences in RDF due to the loss of small-scale velocity caused by filtering. It is clearly observed that FDNS underpredicts the RDF at St = 0.5 and overpredicts the RDF at St = 2.0. The missing small-scale turbulence has little effects on the RDF at St = 1.0. With the incorporation of the ANN model, the contributions of the missing small-scale velocity to particle clustering, RDF are compensated. The RDF of FDNS + ANN is well consistent with that of DNS. Figure 10 (b) (d) (f) compares the curves of the mean of the absolute RRV as a function of dimensionless distance for DNS, FDNS, and FDNS + ANN at different St numbers. Compared to DNS, FDNS underpredicts the RRV of particles due to the loss of small-scale velocity caused by filtering at all of the 3 Stokes numbers. The inclusion of the ANN model compensates for the small-scale velocity and mitigates the RRV errors between FDNS and DNS at all of the 3 St numbers. Therefore, with the inclusion of the ANN model in FDNS, significant improvements in the statistics of particle relative motions can be achieved.

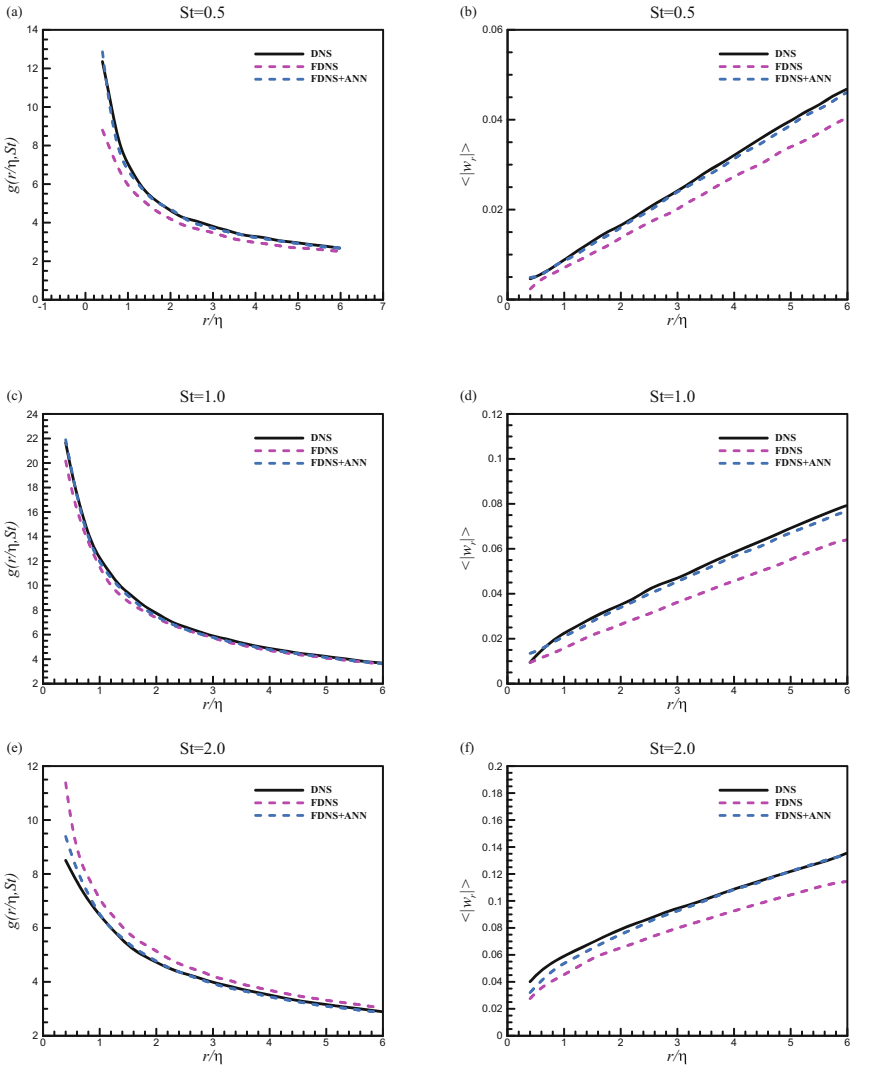


**Fig. 10.** Variation of the RDF [(a) (c) and (e)] and RRV [(b) (d) and (f)] with $r/\eta$ in the DNS, FDNS, and FDNS with ANN model for St = 0.5, 1.0, and 2.0.

## 6 Conclusion

In this study, we have modeled the nonlinear terms and pressure gradient terms in the small-scale turbulence equations using an ANN model. The ANN model is used to calculate these two terms from large-scale information of the flow field. The equation is approximately linearized. In ANN modeling, large-scale flow velocity and velocity gradient tensor are used as inputs for optimization using an error backpropagation algorithm. Combining equations with ANN models can numerically solve turbulent small-scale flows. The study used flow fields with Reynolds numbers of 64.43 and 128.78 to test the

model. By comparing the ANN and DNS data, the ANN model can well maintain the vortex structure of small-scale flow in the prediction of the flow field. Statistically, the probability density distribution of small-scale velocity and velocity gradient of the ANN model and DNS can also be well consistent. Through velocity gradient probability density distribution, ANN data can also reflect the intermittency of small-scale turbulence. Since the ANN model is constructed using FDNS data for small-scale turbulence, it can be seen from the energy spectrum that the ANN model can supplement the small-scale flow field energy filtered out by FDNS during the filtering process. The addition of the ANN model serves as a supplement to the small-scale flow, allowing FDNS to make more accurate predictions of the relative motion of inertial particles. Once the neural network model is modeled, the Reynolds number and resolution for the same flow field remain unchanged, so only one network training is required, and then it can be directly used. This makes this research method a good application prospect in engineering applications. Combining the method of large eddy simulation can not only obtain relatively accurate small-scale flow field data but also save a lot of computational time and cost compared with DNS. The results of this paper show that the method is feasible for constructing small-scale flow fields, and the calculated data are reliable. This study expands the research methods of turbulent small-scale flows and further proves the feasibility of applying ANN modeling to turbulent research.

In future work, we will continue to study how many time-steps this model can recover the missing SGS flow field while ensuring data accuracy. Our research currently focuses only on flows at Reynolds numbers of 64.43 and 128.78, flows at higher Reynolds numbers need to be studied in future work.

**Acknowledgments.** This work was supported by the NSFC Basic Science Center Program for "Multiscale Problems in Nonlinear Mechanics" (Grant No. 11988102), the NSFC Program (Grant No. 12272380), the National Key Project (Grant No. GJXM92579).

## References

1. Lesieur, M., Metais, O.: New trends in large-eddy simulations of turbulence. Annu. Rev. Fluid Mech.. Rev. Fluid Mech. **28**, 45–82 (1996)
2. Meneveau, C., Katz, J.: Scale-invariance and turbulence models for large-eddy simulation. Annu. Rev. Fluid Mech.. Rev. Fluid Mech. **32**(1), 1–32 (2000)
3. Sreenivasan, K.R., Antonia, R.A.: The phenomenology of small-scale turbulence. Annu. Rev. Fluid Mech.. Rev. Fluid Mech. **29**, 435–472 (1997)
4. Ghate, A.S., Lele, S.K.: Gabor mode enrichment in large eddy simulations of turbulent flow. J. Fluid Mech. **903**, A-13 (2020)
5. Pierce, C.D., Moin, P.: Progress-variable approach for large-eddy simulation of non-premixed turbulent combustion. J. Fluid Mech. **504**, 73–97 (2004)
6. Smagorinsky, J.: General circulation experiments with the primitive equations: I the basic experiment. Mon. Weather Rev. **91**(3), 99–164 (1962)
7. Bardina, J., Ferziger, J., et al.: Improved subgrid-scale models for large-eddy simulation. In: Fluid and Plasma Dynamics Conference, 13th, Snowmass, CO, U.S.A. American Institute of Aeronautics and Astronautics (1980)
8. Bardina, J., Ferziger, J., et al.: Improved turbulence models based on large eddy simulation of homogeneous, incompressible turbulent flows. Ph.D. thesis (1983)

9. Clark, R.A., Ferziger, J.H., et al.: Evaluation of subgrid-scale models using an accurately simulated turbulent flow. J. Fluid Mech. **91**(01), 1–16 (1979)
10. Zang, Y., Street, R., et al.: A dynamic mixed subgrid-scale model and its application to turbulent recirculating flows. Phys. Fluids A **5**, 3186–3196 (1993)
11. Germano, M.: Turbulence: the filtering approach. J. Fluid Mech. **238**, 325–336 (1992)
12. Germano, M., Piomelli, U., et al.: A dynamic subgrid-scale eddy viscosity model. Phys. Fluids A **3**, 1760–1765 (1990)
13. Vreman, B., Geurts, B., et al.: Large-eddy simulation of the temporal mixing layer using the Clark model. Theoret. Comput. Fluid Dyn.. Comput. Fluid Dyn. **8**(4), 309–324 (1996)
14. Morinishi, Y., Vasilyev, O.V.: A recommended modification to the dynamic two-parameter mixed subgrid scale model for large eddy simulation of wall bounded turbulent flow. Phys. Fluids **13**(11), 3400–3410 (2001)
15. Yang, Z., Cui, G., et al.: A modified nonlinear sub-grid scale model for large eddy simulation with application to rotating turbulent channel flows. Phys. Fluids **24**(7), 075113 (2012)
16. Yang, Z., Cui, G., et al.: Large eddy simulation of rotating turbulent channel flow with a new dynamic global-coefficient nonlinear subgrid stress model. J. Turbul.Turbul. **13**, 1–20 (2012)
17. Ghaisas, N., Frankel, S.: Dynamic gradient models for the sub-grid scale stress tensor and scalar flux vector in large eddy simulation. J. Turbul.Turbul. **17**, 30–50 (2016)
18. Ghosal, S., Lund, T.S., et al.: A dynamic localization model for large-eddy simulation of turbulent flows. J. Fluid Mech. **286**, 229–255 (1995)
19. Zhou, Z., Wang, S., et al.: A structural subgrid-scale model for relative dispersion in large-eddy simulation of isotropic turbulent flows by coupling kinematic simulation with approximate deconvolution method. Phys. Fluids **30**(10), 105110 (2018)
20. Jin, G., He, G.: A nonlinear model for the subgrid timescale experienced by heavy particles in large eddy simulation of isotropic turbulence with a stochastic differential equation. New J. Phys. **15**(3), 1–27 (2013)
21. Mcculloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. J. Symb. Log.Symb. Log. **5**, 115–133 (1943)
22. He, Q., Li, N., et al.: A survey of machine learning algorithms for big data. Moshi Shibie yu Rengong Zhineng/Pattern Recogn. Artif. Intell. **27**(4), 327–336 (2014)
23. LeCun, Y., Bengio, Y., et al.: Deep learning. Nature **521**(7553), 436–444 (2015)
24. Hornik, K., Stinchcombe, M., et al.: Multilayer feedforward networks are universal approximators. Neural Netw.Netw. **2**, 359–366 (1989)
25. Brunton, S., Noack, B., et al.: Machine learning for fluid mechanics. Annu. Rev. Fluid Mech.. Rev. Fluid Mech. **52**, 477–508 (2020)
26. Yarlanki, S., Rajendran, B., et al.: Estimation of turbulence closure coefficients for data centers using machine learning algorithms. In: IEEE 2012 13th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm), San Diego, CA, USA, pp. 38–42 (2012)
27. Wang, J., Wu, J., et al.: Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data. Phys. Rev. Fluids **2**, 034603 (2017)
28. Fukami, K., Fukagata, K., et al.: Super-resolution reconstruction of turbulent flows with machine learning. J. Fluid Mech. **870**, 106–120 (2019)
29. Qu, J., Tang, Z., et al.: Turbulence feature extraction and analysis based on convolutional neural networks. In: The 10th National Conference on Fluid Mechanics (2018). (in Chinese)
30. Gamahara, M., Hattori, Y.: Searching for turbulence models by artificial neural network. Phys. Rev. Fluids **2**(5), 054604 (2016)
31. Sarghini, F., Felice, G., et al.: Neural networks based subgrid scale modeling in large eddy simulations. Comput. Fluids. Fluids **32**(1), 97–108 (2003)

32. Xie, C., Wang, J., et al.: Spatial artificial neural network model for subgrid-scale stress and heat flux of compressible turbulence. Theor. Appl. Mech. Lett.. Appl. Mech. Lett. **10**(1), 27–32 (2020)
33. Zhou, Z., He, G., et al.: Subgrid-scale model for large-eddy simulation of isotropic turbulent flows using an artificial neural network. Comput. Fluids. Fluids **195**, 104319 (2019)
34. Beck, A., Flad, D., et al.: Deep neural networks for data-driven LES closure models. J. Comput. Phys.Comput. Phys. **398**, 108910 (2019)
35. Milano, M., Koumoutsakos, P.: Neural net modeling for near wall turbulent flow. J. Comput. Phys.Comput. Phys. **182**(1), 1–26 (2002)
36. Yin, Y., Li, H., et al.: Application of machine learning assisted turbulence modeling in flow separation prediction. Acta Aerodynamica Sinica **39**(2), 23–32 (2021)
37. Xie, C., Yuan, Z., et al.: Artificial neural network-based subgrid-scale models for large-eddy simulation of turbulence (in Chinese). Lixue Xuebao/Chin. J. Theor. Appl. Mech. **53**(1), 1–16 (2021)
38. Hinze, J.O.: Turbulence: An Introduction to Its Mechanism and Theory. McGraw-Hill Book Company, inc., New York (1960)
39. Zhang, Z., Cui, G., et al.: Theory and Modeling of Turbulence. 2nd edn. Tsinghua University Press, Beijing (2017)
40. Yan, X., Li, J., et al.: The influence of sub-grid scale motions on particle collision in homogeneous isotropic turbulence. Acta Mech. Sin. **34**(1), 22–36 (2018)
41. Jiao, L., Yang, S., et al.: Seventy years beyond neural networks: retrospect and prospect. JisuanjiXuebao (Chin. J. Comput.) **39** (2016)
42. Rumelhart, D., Williams, R., et al.: Learning representations by back-propagating errors. Nature **323**, 533–536 (1986)
43. Liu, B., Tang, J., et al.: Deep learning methods for super-resolution reconstruction of turbulent flows. Phys. Fluids **32**(2), 025105 (2020)
44. Jeong, J., Hussain, F.: On the identification of a vortex. J. Fluid Mech. **285**, 69–94 (1995)
45. Zhou, Z.: Large-eddy simulation of particle-laden isotropic turbulent flows and sub-grid scale models (2019)
46. Ray, B., Collins, L.R.: A subgrid model for clustering of high-inertia particles in large-eddy simulations of turbulence. J. Turbul.Turbul. **15**(6), 366–385 (2014)
47. Sundaram, S., Collins, L.R.: Collision statistics in an isotropic particle-laden turbulent suspension .1. Direct numerical simulations. J. Fluid Mech. **335**, 75–109 (1997)