energies

MDPI

Article

A Dataset Establishment Method for Wind Turbine Wake and a Data-Driven Model of Wake Prediction

Qinghong Tang [1], Yuxin Wu [1,2,*], Changhua Li [1], Peiyao Duan [1], Jiahao Wu [1] and Junfu Lyu [1]

[1] Department of Energy and Power Engineering, Tsinghua University, Beijing 100084, China
[2] Institute for Carbon Neutrality, Tsinghua University, Beijing 100084, China
* Correspondence: wuyx09@mail.tsinghua.edu.cn

Abstract

A cross-construction method is proposed to establish a wind turbine wake dataset with significantly reduced computational fluid dynamics (CFD) costs. This method involves adjusting one operating parameter, such as the tip speed ratio (TSR), while maintaining the others at their optimal values. This procedure is repeated across another parameter (inflow velocity) to generate a sparse but informative dataset. CFD simulations were performed using large eddy simulation (LES) coupled with an actuator line model (ALM) to generate data. A pre-training and fine-tuning network based on error classification (PFNEC) was developed, achieving high prediction accuracy with coefficients of determination of 0.9750 and 0.9851 for two validation conditions. Two models based on a softmax function and a residual block were designed, and they achieved the best performance, with coefficients of determination of 0.9921 and 0.9891 under different conditions. The Fourier embedding was applied to enhance input features of neural networks. Four samples added to the original dataset improved the prediction accuracy for extreme operating conditions, from coefficient of determination values of 0.7143 and 0.7034 to 0.9939 and 0.9886 with Fourier embedding. This cross-construction method can significantly reduce the cost of dataset establishment. The models exhibited reliable generalization and prediction accuracy.

Keywords: wind turbine wake prediction; neural network; error classification learning; softmax function; residual block

1. Introduction

As a form of clean energy, wind power has become an increasingly significant contributor to the energy supply [1]. A wind turbine is an energy-conversion device that converts wind energy into electricity and can be classified into horizontal axis wind turbines (HAWTs) and vertical axis wind turbines (VAWTs) [2], and other configurations. Wind farms are the primary configuration for large-scale wind energy utilization. The wind turbine wake posed a major challenge to operational optimization and the maximization of energy production [3]. It is important to predict wake characteristics and provide more wake information for the control strategy.

In the early stage of wind turbine wake prediction, the analytical model was carried out by researchers to perform the prediction. The Jensen model was given in 1983 [4], and then the Gaussian model was proposed with an assumption of a Gaussian profile of wake velocity deficit [5]. A three-dimensional wake model was developed for offshore wind turbines [6]. Analytical models can efficiently estimate wake velocity, but their accuracy remains limited, especially under non-optimal conditions.

Check for updates

Academic Editor: Francesco Castellani

Received: 25 January 2026
Revised: 28 February 2026
Accepted: 6 March 2026
Published: 9 March 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

The machine learning method was also applied to the prediction of wind turbine wakes. Neural networks can represent the nonlinear mapping between operating conditions and wake quantities, and they have become widely used for flow prediction [7]. The neural network architectures employed in the prediction of wind turbine wake fields include multilayer perceptrons (MLPs), also referred to as feedforward artificial neural networks (ANNs), convolutional neural networks (CNNs), and graph neural networks (GNNs) [8]. In addition, more advanced frameworks have recently been explored, such as physics-informed neural networks (PINNs) [8]. Wake predictions are performed using the wind speed, tip speed ratio (TSR), and turbulence intensity as the input features. The wind velocity of a single turbine was predicted by the MLP, and the data was generated by computational fluid dynamics (CFD) [9]. The turbulence intensity and wake velocity of a standalone wind turbine with uniform inflow were predicted by the ANN trained by CFD simulations [10]. Ti et al. [11,12] adopted an ANN for wake and power prediction, and the results were acceptable. Nakhchi et al. [13] established a dataset by simulation and also used an ANN architecture to predict the wake field with low deviation. A surrogate model trained on real-world light detection and ranging (LiDAR) was used to extract latent space and forecast wake flows [14]. To incorporate with the analytical wake model, the ANN was used to predict nonlinear wake expansion and calculate wake flows with the Jensen model [15]. To more effectively capture spatial correlations in wake flow fields, the convolutional neural network (CNN) was also applied to spatiotemporal wake prediction [16,17]. A deep convolutional conditional generative adversarial network (DC-CGAN) was proposed. It can capture wake deflection with the turbine yaw angle, and the CGAN has the capability of generalization [18]. LES data were used to train the CNN autoencoder to reconstruct the time-averaged flow field [19]. A Graph Neural Operator (GNO) architecture containing two GNN layers was designed, which can identify zones characterized by strong wake interactions [20]. The physics-informed neural network (PINN) was proposed, and the governing equations can be added into the loss function to make neural networks learn physical law [21]. For wind turbine wake prediction, the measurement data was considered as data loss to reconstruct multi-turbine wake fields [22]. The actuator disk model (ADM) was added in the governing equations to calculate the force acting on the fluid by the wind turbine, and to calculate the wake field [23]. A multi-scale prediction framework of wind turbine wake field was constructed and had high accuracy for single turbine, turbine array, and utility-scale wind farm [24].

Different neural network architectures have been developed for wind turbine wake prediction, but the design of these models does not explicitly consider the intrinsic physical characteristics of turbine wake flows. In this study, the features of wind turbine wakes are incorporated into both data feature engineering and neural network design to improve predictive performance.

The neural networks are capable of rapid wake prediction, but effective training requires a high-quality dataset. 403 simulations were conducted to generate the dataset, and then an artificial neural network (ANN) was established to predict the wake flows [11]. Nakhchi et al. [13] conducted a series of high-fidelity simulations using LES to generate training data and adopted the Extreme Gradient Boosting machine learning algorithm for wake prediction, and they found the Extreme Gradient Boosting model was more accurate than the ANN model. Li et al. [17] conducted 17 temporal simulations under different turbine yaw angles as training data, and a convolutional neural network was trained to infer flow fields. A combination of an engineering wake model and LES simulations was used to train a data-driven model with reduced computational cost [25]. Previous studies typically identify a set of wind turbine operating parameters, vary each parameter uniformly, and then combine all parameter levels to form a dense parameter-space dataset.

This approach is computationally expensive because it requires a full CFD simulation for every parameter combination. Therefore, there is a need for approaches that can efficiently construct a comprehensive dataset at a lower cost.

This study addresses two fundamental questions: (1) How can a comprehensive dataset be constructed in a cost-effective manner? (2) How can wake characteristics be effectively embedded into neural network architectures? A neural network based on error classification was designed to improve predictive performance. The Fourier embedding method was introduced to enrich the input features [26]. This study proposed a cross-construction method for dataset establishment, and a sparse wind turbine wake dataset was built at a low cost by using this method. The neural networks were designed for wind turbine wake prediction, and it is demonstrated that the Fourier embedding can improve predictive accuracy. In Section 2, CFD setup and validation are described. The comparison between a fully connected network (FCN) and an artificial neural network (ANN), and the introduction of the cross-construction method, are presented in Section 3. The advanced models based on error classification are illustrated, and results are analyzed in Section 4. The comprehensive results and discussion are in Section 5. The generalization of neural networks and the modification of the cross-construction dataset are in Section 6. The conclusions of this study are in Section 7.

2. CFD Framework

The filtered continuity equation and the filtered Navier–Stokes equation for the LES method in incompressible flows were shown as follows [27–29]:

$$\frac{\partial \overline{u}_i}{\partial x_i} = 0 \tag{1}$$

$$\frac{\partial \overline{u}_i}{\partial t} + \frac{\partial}{\partial x_j}\left(\overline{u}_i \overline{u}_j\right) = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \frac{\mu}{\rho}\frac{\partial^2 \overline{u}_i}{\partial x_j \partial x_j} - \frac{\partial \tau_{ij}}{\partial x_j} + f_i \tag{2}$$

where $\overline{u}_i$ is the filtered velocity tensor, the overbar represents grid-filtered variables, p is pressure, ρ is air density 1.225 kg/m^3, μ is dynamic viscosity, and f_i is the external force calculated by ALM. $\tau_{ij} = \overline{u_i u_j} - \overline{u_i}\,\overline{u_j}$ is the sub-grid scale (SGS) stress tensor. In the One-Equation model:

$$\tau_{ij} - \frac{1}{3}\tau_{kk}\delta_{ij} = -2\nu_{sgs}\overline{S}_{ij} \tag{3}$$

$$\overline{S}_{ij} = \frac{1}{2}\left(\frac{\partial \overline{u}_i}{\partial x_j} + \frac{\partial \overline{u}_j}{\partial x_i}\right) \tag{4}$$

$$\frac{\partial k_{sgs}}{\partial t} + \overline{u}_j \frac{\partial k_{sgs}}{\partial x_j} = -\left(\tau_{ij} - \frac{1}{3}\delta_{ij}\tau_{kk}\right)\overline{S}_{ij} - C_\varepsilon \frac{k_{sgs}^{3/2}}{\Delta} - \varepsilon_w + \frac{\partial}{\partial x_j}\left[\left(C_k \Delta \sqrt{k_{sgs}} + \nu\right)\frac{\partial k_{sgs}}{\partial x_j}\right] \tag{5}$$

$$\nu_{sgs} = C_k \Delta \sqrt{k_{sgs}} \tag{6}$$

where ν_{sgs} is sub-grid kinematic viscosity, C_ε and C_k are two constant coefficients where $C_\varepsilon = 0.93$ and $C_k = 0.0673$, Δ is the filter size, and ε_w is the near-wall dissipation correction.

The LES One-Equation model was implemented in this simulation [30,31]. As for the wind turbine wake flow, the ALM was coupled with the LES method for wind turbine flow simulation [32]. The wind turbine model and experimental data come from NTNU (Norges Teknisk-Naturvitenskapelige Universitet) "Blind test 1" [33,34]. The computational domain is depicted in Figure 1. The size of the computational domain is identical to that of the experiment in height and width. The inflow velocity $U_0 = 10$ m/s is set as an inlet boundary, and the outlet is set as a pressure boundary. The turbulence intensity of the

inlet is 0.3%, which is consistent with the experimental setup. To avoid resolving the wall boundary layers with a finely refined mesh, slip boundary conditions are applied on the top, bottom, front, and back walls. All of the simulations were conducted using the open software SOWFA (Simulator fOr Wind Farm Applications).

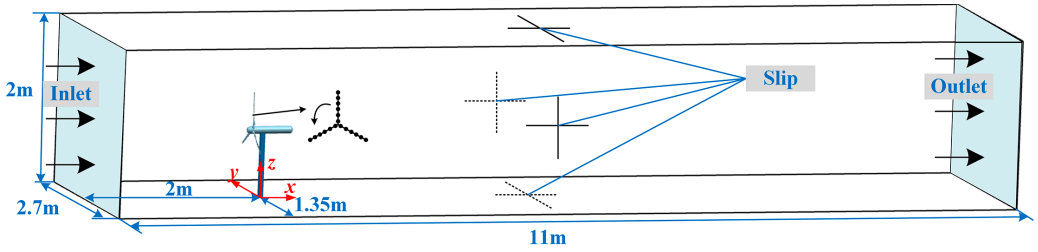


Figure 1. The computational domain.

The initial mesh size of the mesh region is 0.15 m for the entire computational domain. The wake region was refined progressively. The first refinement region is $x \in [-1.8\text{ m}, 9\text{ m}]$, $y \in [-1.35\text{ m}, 1.35\text{ m}]$, and $z \in [0\text{ m}, 2\text{ m}]$. The second refinement region is $x \in [-0.5\text{ m}, 4.9\text{ m}]$, $y \in [-0.9\text{ m}, 0.9\text{ m}]$, and $z \in [0\text{ m}, 1.5\text{ m}]$. The third refinement region is $x \in [-0.3\text{ m}, 4.7\text{ m}]$, $y \in [-0.9\text{ m}, 0.9\text{ m}]$, and $z \in [0\text{ m}, 1.5\text{ m}]$. The fourth refinement region is $x \in [-0.2\text{ m}, 2.8\text{ m}]$, $y \in [-0.9\text{ m}, 0.9\text{ m}]$, and $z \in [0.3\text{ m}, 1.5\text{ m}]$. The grid size in the finest region is 0.009375 m. The unsteady time step corresponded to a 1° rotor rotation across all TSR conditions, with a mean flow field averaging window exceeding 50 revolutions to ensure statistical convergence. The inflow turbulence intensity was set as 0.3% at the inlet, which is the same as the experimental setup [34].

In the application of ALM, the Gaussian projection function is used to make the distribution of lift force and drag force smoother [33]:

$$\eta_\varepsilon = \frac{1}{\varepsilon^3 \pi^{3/2}} \exp\left(-\frac{d^2}{\varepsilon^2}\right) \tag{7}$$

where η_ε is a smooth projection function, ε is the Gaussian projection width, and d is the distance between the local actuator point and the local grid. The Gaussian projection width is set as $\varepsilon = 0.0018$ m in this study.

The definition of TSR is shown in Equation (8). The validation of the simulation through the comparison to experimental measurements of power coefficient C_p (Equation (9)) and thrust coefficient C_T (Equation (10)) was shown in Figure 2. The numerical results agree well with the experimental data for the power coefficient C_p and thrust coefficient C_T from low TSR to high TSR. The numerical framework of this study is credible for the simulation of wind turbine wake flows and for constructing a wake dataset.

$$\lambda = \frac{\omega R}{U_0} \tag{8}$$

$$C_p = \frac{M\omega}{0.5\rho A U_0^3} \tag{9}$$

$$C_T = \frac{F}{0.5\rho A U_0^2} \tag{10}$$

$$U_d = \frac{U_0 - U_{\text{mean}}}{U_0} \tag{11}$$

where λ is TSR, ω is rotational speed (rad/s), R is radius of rotor (m), U_0 is free stream velocity 10 m/s, U_{mean} is the time-averaged velocity (m/s), M is torque of turbine (N·m), F is axial force of turbine (N), A is swept area of turbine (m^2).

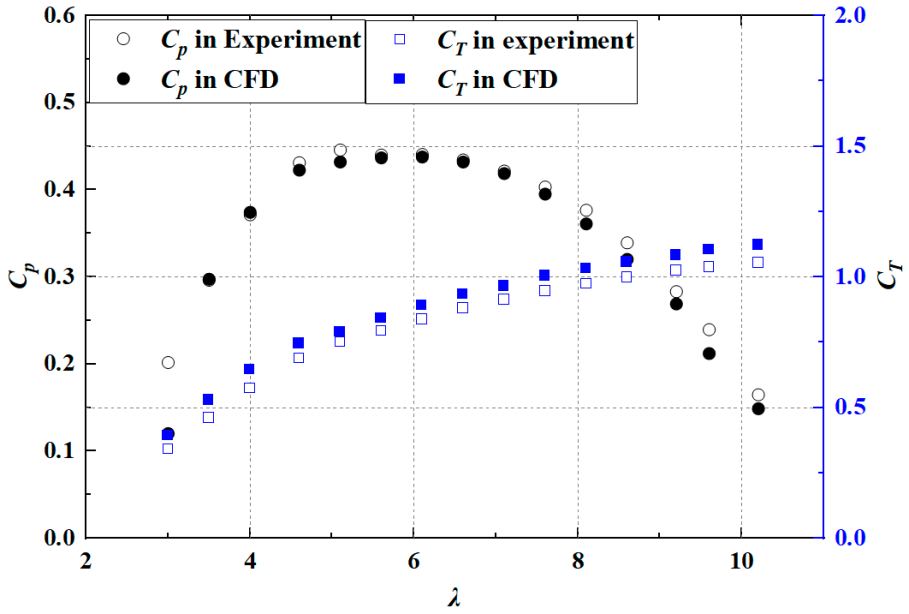


Figure 2. Power and thrust coefficient.

3. Verification of Fully Connected Network and Dataset Establishment

3.1. Comparison Between Artificial Neural Network and Fully Connected Network

CFD simulations were performed for different conditions by cross-varying of TSR λ = 5.6 and design wind speed U_0 = 10 m/s to build a dataset. Table 1 lists variation in input variables (λ, U_0) from Cases 1–36. Cases 35 and 36 are used as validation cases. The CFD dataset was generated on Intel(R) Xeon(R) CPU E5-2678 v3 @ 2.50 GHz using parallel processing. The training and inference of the neural networks were performed on a CPU 13th Gen Intel(R) Core(TM) i7-13700k. In each training, the models were trained for 500 epochs to guarantee convergence. The code link is as follows: https://github.com/CAME-THU/Wake-prediction-model-PFNEC.

Table 1. The conditions of the dataset.

Case	λ	U_0 (m/s)	Case	λ	U_0 (m/s)
1	5.6	6	19	5.6	15
2	5.6	6.5	20	3	10
3	5.6	7	21	3.5	10
4	5.6	7.5	22	4	10
5	5.6	8	23	4.6	10
6	5.6	8.5	24	5.1	10
7	5.6	9	25	6	10
8	5.6	9.5	26	6.1	10
9	5.6	10	27	6.6	10
10	5.6	10.5	28	7.1	10
11	5.6	11	29	7.6	10
12	5.6	11.5	30	8.1	10

Table 1. *Cont.*

Case	λ	U_0 (m/s)	Case	λ	U_0 (m/s)
13	5.6	12	31	8.6	10
14	5.6	12.5	32	9.2	10
15	5.6	13	33	9.6	10
16	5.6	13.5	34	10.2	10
17	5.6	14	35	7.3	11.5
18	5.6	14.5	36	4.0	8.5

An ANN was applied to wind turbine wake prediction, and the model was decomposed into multiple sub-models [11–13]. The architecture of the ANN is shown in Figure 3. A total of 1017 sub-models were designed in this study, and one sub-model contains 75 elements of output U, and the 75 elements are consistent with the setting in Ref [13]. The hidden layer of each sub-model has 10 neurons. The performance of the ANN is compared with neural networks designed in this study.

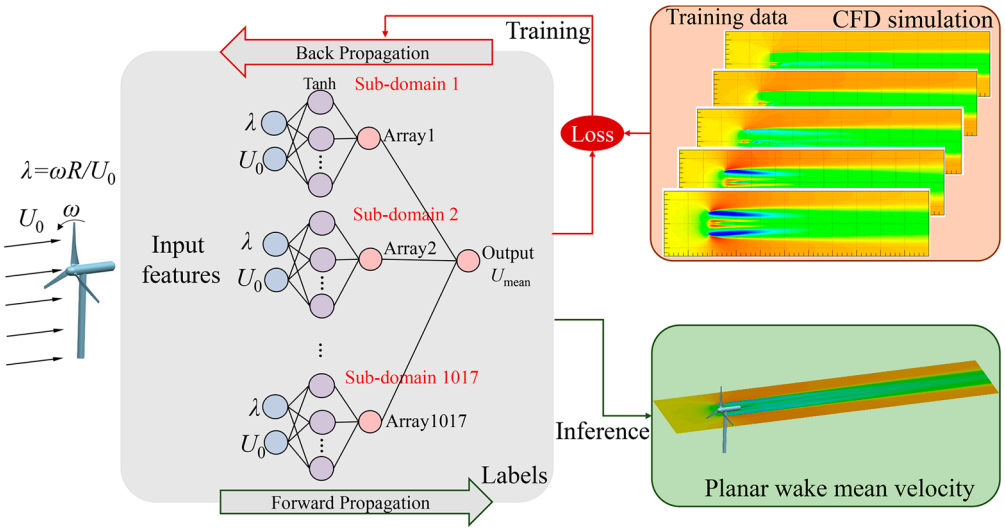


Figure 3. Architecture of ANN.

The architecture of the FCN used in this study is designed and shown in Figure 4. The inputs of FCN are TSR λ and freestream wind speed U_0, and the output of FCN is grid-point velocities of the XY plane at hub height z = 0.817 m, which contains n = 76,230 spatial points. In order to obtain better prediction performance and avoid overfitting, two hidden layers with 128 and 256 neurons in FCN are selected, respectively.

The relative error δ between neural network (NN) predictions and CFD simulations is defined as:

$$\delta = \frac{U_{\text{mean}-\text{NN}} - U_{\text{mean}-\text{CFD}}}{U_{\text{mean}-\text{CFD}}} \times 100 \tag{12}$$

where $U_{\text{mean-CFD}}$ is the mean velocity U_{mean} calculated by CFD, and the $U_{\text{mean-NN}}$ is the mean velocity regressed by the neural network algorithm.

The coefficient of determination R^2 and root mean square error (RMSE) are defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{N} (y_i - \overline{y})^2} \tag{13}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|^2} \tag{14}$$

where N is the number of samples, and y_i is the value of the CFD results, and $\hat{y}_i$ is the value of NN results, and $\overline{y}$ is the average value of CFD data.

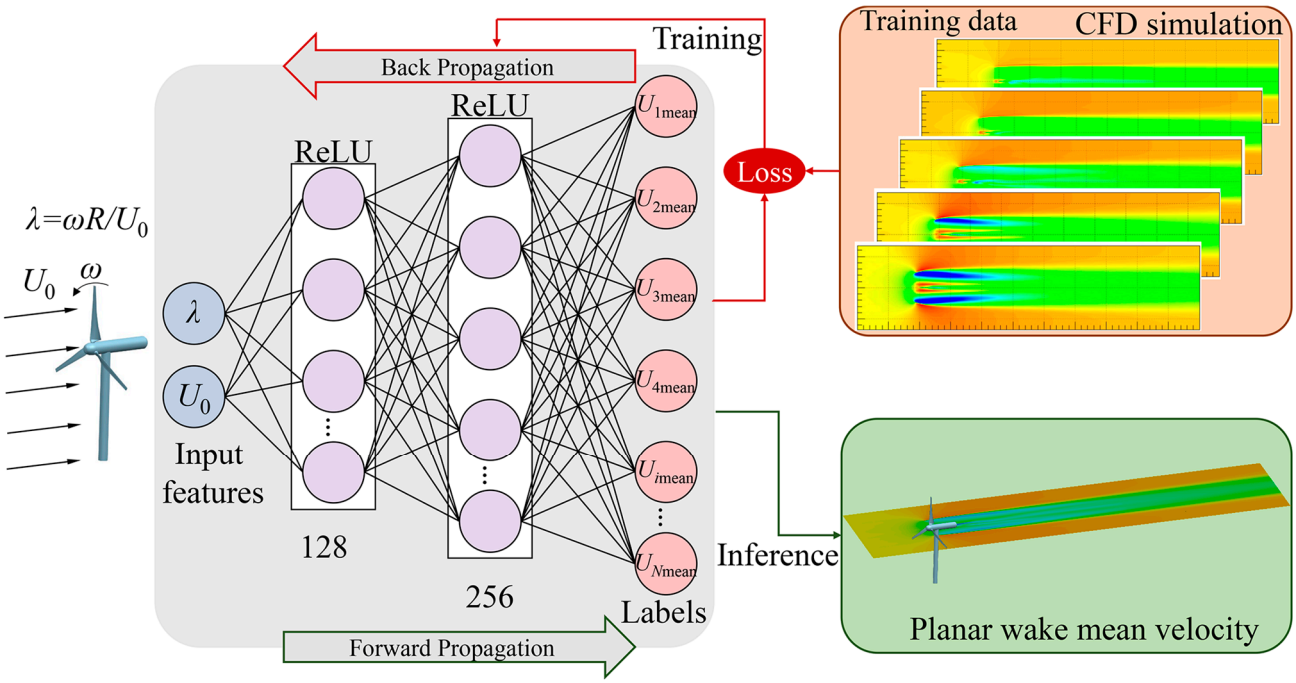


Figure 4. Architecture of FCN.

In order to compare the predictive ability of ANN and FCN for external conditions (λ, U_0) = (7.3, 11.5) and (λ, U_0) = (4.0, 8.5), the inference results are shown in Figure 5. The U_{mean} contour of ANN and FCN is shown in Figure 6. The scatter distributions of the ANN and FCN predictions with respect to the CFD (ground-truth) are depicted, and ±5% as well as ±10% relative error bands are also marked in Figure 5. Based on the comparisons in Figures 5 and 6, the FCN consistently exhibits better performance than the ANN.

The FCN performs better than the ANN for the (λ, U_0) = (4.0, 8.5) condition. The coefficient of determination R^2 and RMSE for (λ, U_0) = (7.3, 11.5) are R^2 = 0.9541 and RMSE = 0.5366, which is a little larger than for (λ, U_0) = (4.0, 8.5) with R^2 = 0.9505. The prediction results of the ANN are R^2 = 0.7374 for (λ, U_0) = (7.3, 11.5) and R^2 = 0.3858 for (λ, U_0) = (4.0, 8.5). There is a large difference between the ANN and CFD in Figure 6. In the study by Ti et al. [11], the RANS method was used to generate data for training. The flow data were interpolated into $x \times y \times z = 40 \times 250 \times 24$. Nakhchi et al. [13] used the LES method to generate training data, and the flow field data were interpolated into 360,000 data points for training, but the number of grid elements is 10.36 million for LES. If grid-point data were interpolated into a sparse grid, the detailed features such as sharp gradients and high-frequency structures in the wake would be missed. But in this study, the LES simulation was used to generate training data, and the grid-point data was directly exported without any sparse interpolation. Therefore, both the training and validation results preserve the LES spatial resolution.

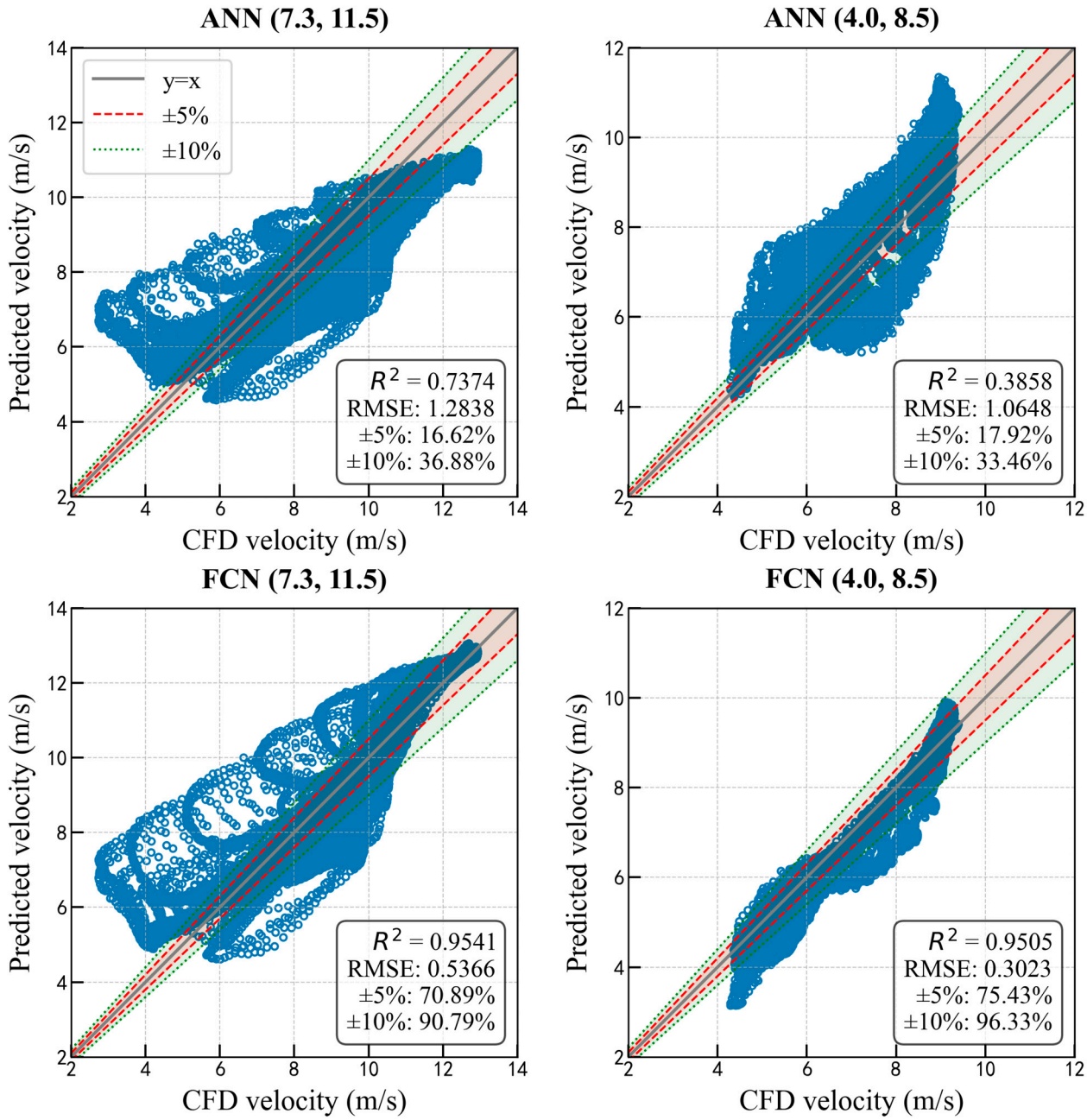


Figure 5. The distribution of mean velocity U_{mean} regressed by ANN and FCN against CFD simulation results.

3.2. Cross-Construction Method

As for a data-driven model, it is important to construct a comprehensive and high-quality dataset. As analyzed in the previous section, high-fidelity simulations are required, and the data should be exported directly without any interpolation to preserve high spatial resolution for training the neural network. However, the computational cost of constructing a high-fidelity CFD dataset is high. In previous studies, researchers established a dataset comprising several hundred cases. Ti et al. [11] established a wake dataset with a total of 403 simulations across a range of varying velocity and turbulence intensity. Therefore, a critical question arises as to how a sparse dataset can be constructed while still retaining the ability to capture comprehensive wake characteristics.

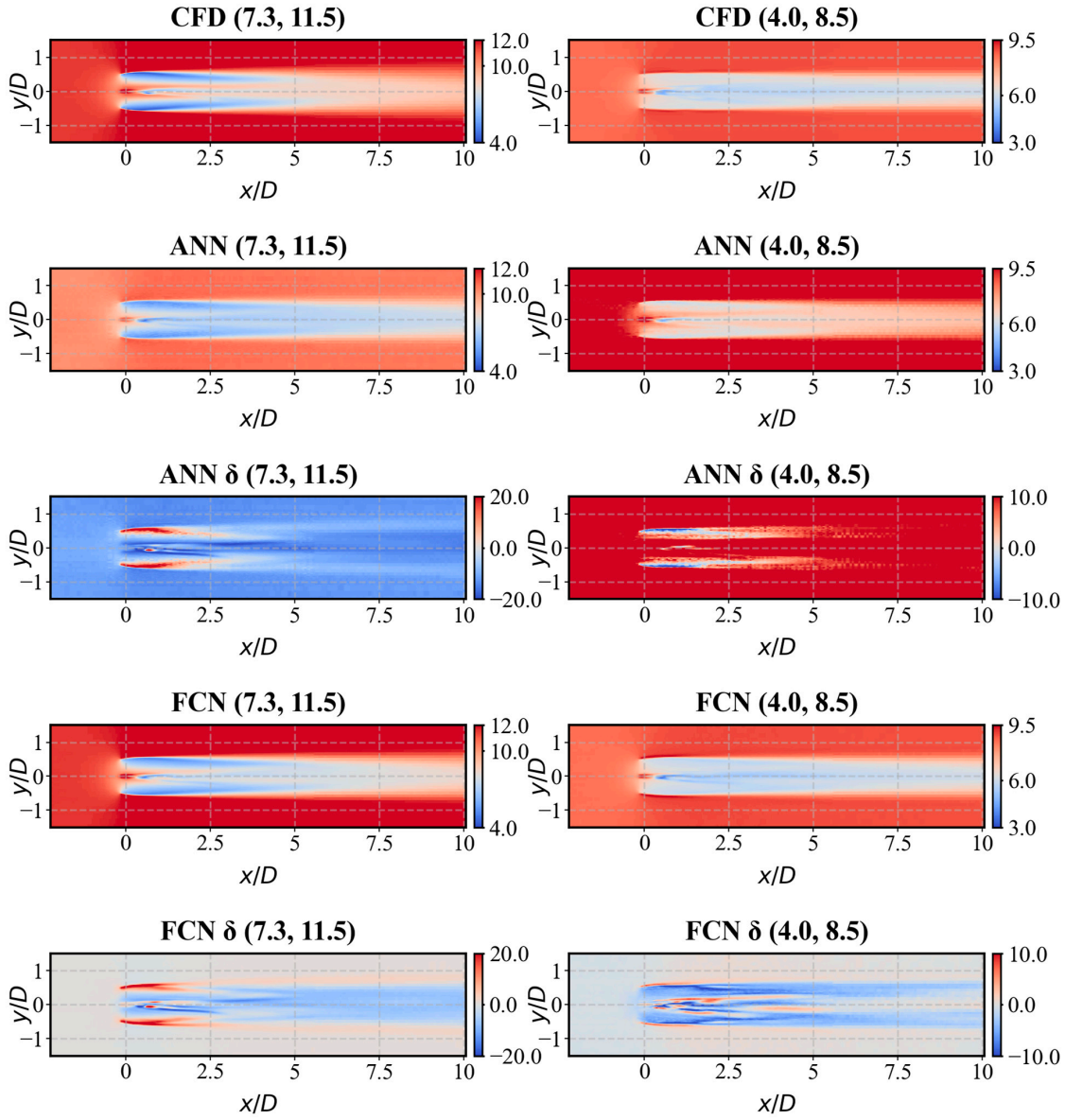


Figure 6. The comparison of ANN and FCN results of mean velocity Umean on the XY plane.

A cross-construction method was proposed to address the above question. Generally, a large number of conditions (blue and black points) should be simulated by CFD to establish a full condition dataset, as shown in Figure 7. In order to reduce computational cost and guarantee diversity of the dataset, a cross-construction sampling method was put forward to establish the dataset, as shown in Figure 8. The parameters of TSR = 5.6 and U_0 = 10 m/s are chosen based on turbine design. The turbine is designed for a wind speed of 10 m/s and a design TSR of λ = 6.0 [33]. When the wind speed is high, the pitch control is employed to reduce TSR to prevent overspeed and protect the turbine components. Accordingly, the parameters of U_0 = 10 m/s and λ = 5.6 are selected to generate the dataset using the cross-construction method in this study. Two parameters are varied in a crossed manner to construct a sparse dataset, as distributed in Figure 8 and in Table 1. Two external conditions were simulated as validation samples, aiming to verify the effectiveness of the cross-construction method. Obviously, it is a sparse dataset with two parameters varying from low to high values. The simulation of blue data points in Figure 7 was avoided. Approximately 270 CFD simulations corresponding to the blue markers in

Figure 7 were avoided using the cross-construction method. The validation is performed to verify whether this dataset can reflect comprehensive characteristics.

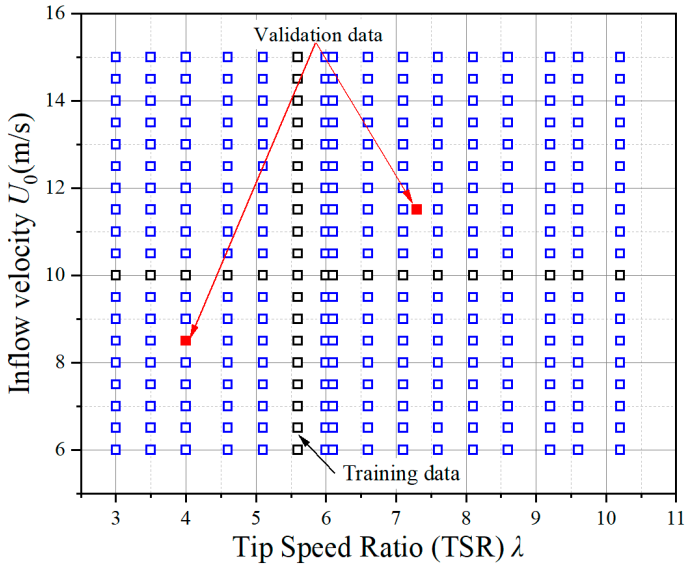


Figure 7. Distribution of parameters TSR and U_0 for constructing the training dataset.

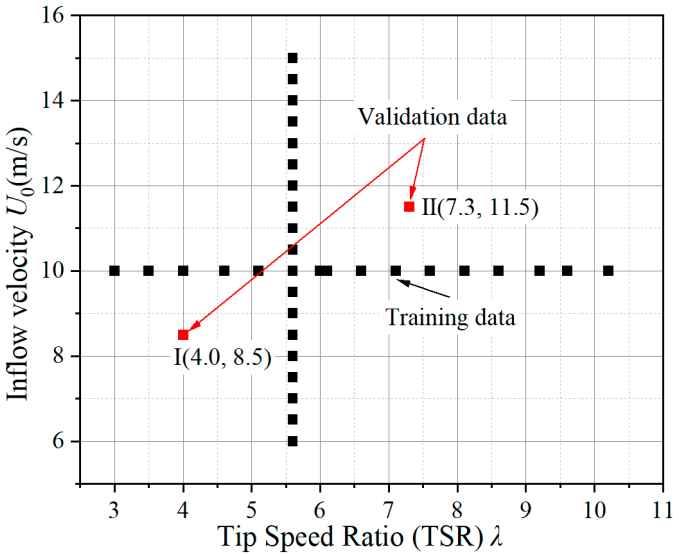


Figure 8. Sparse distribution of parameters TSR and U_0 of the cross-constructed training dataset.

4. Network Design and Analysis

4.1. Fully Connected Network with Coordinate Information

Moreover, it is also necessary to modify the neural network architecture to enable high-accuracy predictions when a model is trained on a sparse dataset. As shown in Figures 5 and 6, it has a large difference in velocity distribution at different spatial locations within the wake flow field. A high-error region occurs near the blade tips. The location near the blade tip has a large velocity deficit due to the effect of the turbine blades. In order to improve the prediction accuracy of the wind turbine wake velocity field, coordinates are included as input features to represent spatial features. The consideration of coordinate information aims to enhance the spatial correlation captured by the neural network. A fully connected network with coordinate (FCN-C) was designed as shown in Figure 9. In FCN-C, the number of inputs is 4, and the output is 1, and every sample has a unique coordinate

value (x, y). As for one operating condition, all of the samples in the 2D plane have the same λ and U_0. The output is the velocity U_{mean} of one spatial point. One sample (x, y) corresponds to one output velocity U_{mean}. Every case has n = 76,230 points in a 2D plane, so that in FCN-C, every case has n = 76,230 samples.

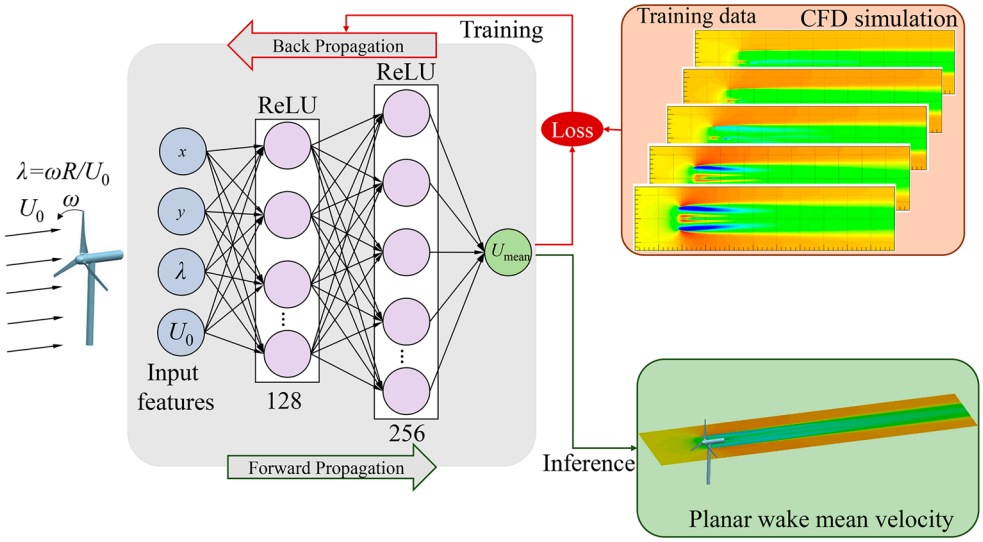


Figure 9. Architecture of a fully connected network with coordinate information.

After consideration of coordinate information, the results of FCN-C are shown in Figures 10 and 11. The predictive result is shown in Figure 10 by scatter plot. The contour of CFD U_{mean}, the predictions, and the corresponding errors are shown in Figure 11. The results show that the prediction accuracy is improved by FCN-C. The coefficient of determination for (λ, U_0) = (7.3, 11.5) is R^2 = 0.9752, which is larger than R^2 = 0.9541 by the FCN model. 93.79% of the n = 76,230 data points have relative error within 10% ($-10\% \leq \delta \leq 10\%$) for ($\lambda$, U_0) = (7.3, 11.5) condition. The coefficient of determination R^2 for (λ, U_0) = (4.0, 8.5) is R^2 = 0.9528, which is also larger than the FCN result R^2 = 0.9505. But 93.18% of the data points, n = 76,230, have relative error within 10% ($-10\% \leq \delta \leq 10\%$), which is lower than the FCN model. The RMSE decreases to 0.3945 and 0.2952 for the two conditions. It is evident in Figure 11 that the high error region near the blade tips is reduced in FCN-C compared with FCN.

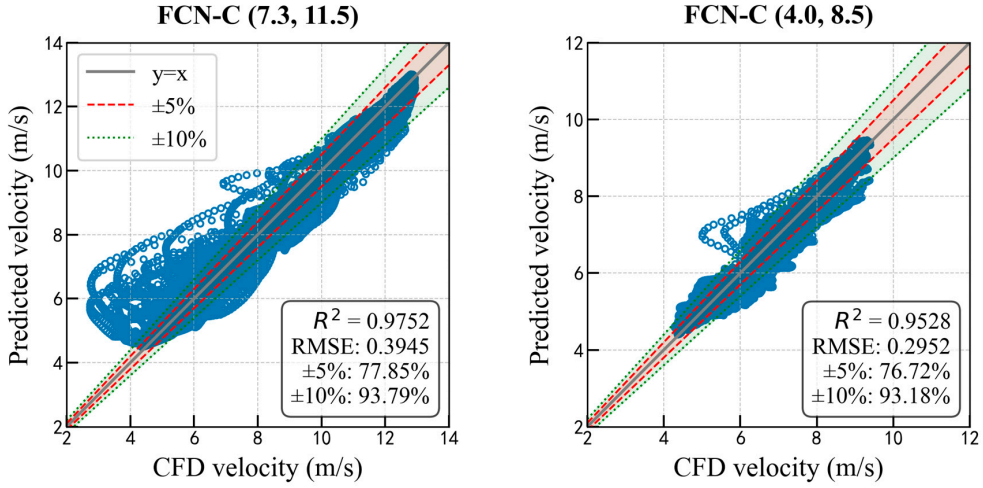


Figure 10. The distribution of mean velocity U_{mean} regressed by FCN-C against CFD simulation results.

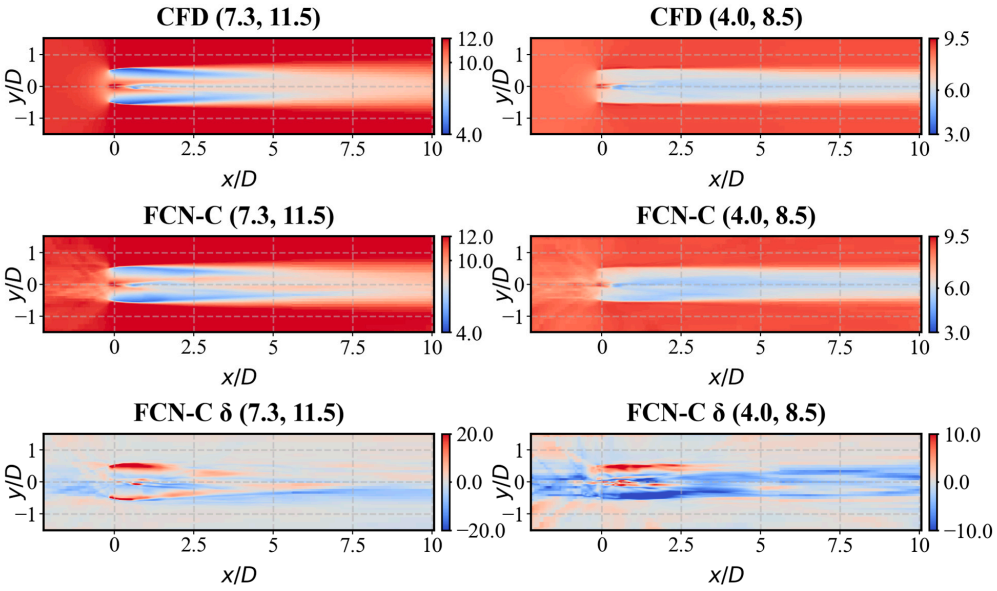


Figure 11. The comparison of FCN-C and CFD simulation results of mean velocity U_{mean} on the XY plane.

4.2. Advanced Model Based on Error Classification

As for the prediction of grid resolution required by LES simulation, a part of the spatial grid points have large deviations from the ground-truth data, especially in low-velocity regions. A fine-tuning neural network was designed to learn features in low-velocity regions. A pre-training and fine-tuning network based on error classification (PFNEC) was designed. As shown in Figure 12, the first neural network is trained initially, and the preliminary network is obtained. The trained preliminary network is used to infer from training data, and the errors are computed against the training data. Samples with large errors indicate that they are difficult for the preliminary network to learn. Then, the relative error is used as the criterion to divide the training data into five categories, as in Figure 12. Bengio et al. proposed the strategy of curriculum learning, starting from easier examples to more difficult examples [35]. In this research, the first training of the network was performed on samples with errors < 10%. After first training with an error < 10%, the NN_1 is obtained, and then the data with 10% $\leq$ error < 30% was used to fine-tune NN_1 to obtain NN_2. This procedure is repeated in the same manner, and five models are ultimately obtained after the fine-tuning process with the total classified data.

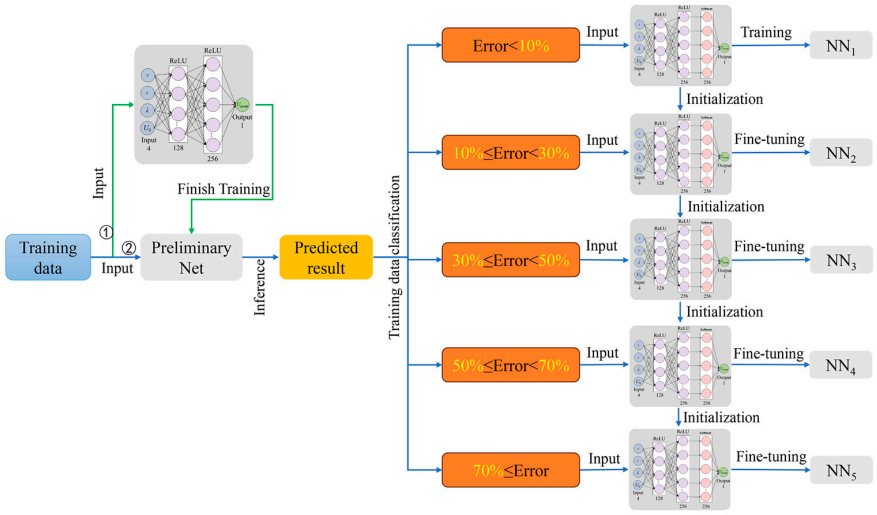


Figure 12. The architecture of pre-training and fine-tuning a network based on error classification.

After training, the five models, NN_1-NN_5, are obtained for different samples according to error performance in the training data. Therefore, as for validation data during inference, the first process is to identify which category the validation sample belongs to, and then the corresponding model is selected to infer. The three classification methods are considered in this study, as shown in Figure 13. (1) The first method is training data classification (TDC), in which the validation samples are classified based on the index of training samples. As for one operating condition for a certain TSR coupled with a certain wind speed, the middle plane contains n = 76,230 spatial points, and corresponding predicted values have different errors, and these points are assigned to different categories. These points are indexed from 0 to 76,229. The validation data is classified according to their index of 76,230 points for one condition, based on which category the training sample with the same index belongs to. (2) The second method is Euclidean distance classification (EDC). As for the FCN-C model, four inputs of coordinates x (i = 0), y (i = 1), TSR (i = 2), and wind speed (i = 3) are used of neural network. The Euclidean distance (in Equation (15)) is used to determine the classification of validation samples. In Equation (15), the dimension is D = 4. As for each testing sample, the Euclidean distance $d(x,y)$ can be calculated for every training sample. In this process, the Euclidean distance related to coordinates x (i = 0) as well as y (i = 1) is calculated first, and the closest indices are found with training samples. The first Euclidean distance discrimination pays more attention to spatial features because high-level error points concentrate in adjacent spatial areas. After the first discrimination, the TSR (i = 2) as the third parameter for Euclidean distance is then incorporated to further determine. Finally, if the previous three parameters i = 0, 1, 2 do not uniquely determine the category of that validation sample, the fourth parameter i = 3 is used to calculate Euclidean distance and identify the nearest neighbor, and assign a unique category. (3) The third method is neural network classification (NNC), in which a classification model is designed to learn based on training error data. The training data is divided into five categories so that five outputs are set for the classification model. After training, the NNC can predict a certain category of validation samples and select and apply the corresponding model for inference. Other classification methods can also be used, such as support vector machine, random forest, and so on.

$$d(x,y) = \sqrt{\sum_{i=1}^{D}(x_i - y_i)^2} \tag{15}$$

where D is the dimension of data space (D = 4 in this study), x is the validation sample, and y is the training sample.

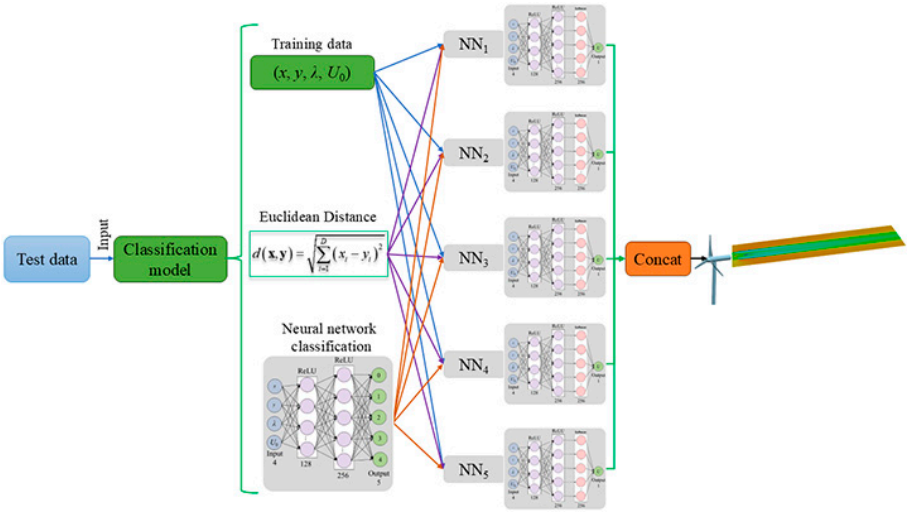


Figure 13. Three classification methods for inference.

4.3. Fully Connected Network with Softmax

Afully connected network with a softmax layer (FCN-S) is designed as shown in Figure 14. A softmax layer is added before the output. The softmax function is often used in classification networks to scale the output satisfying the probability rule with the sum equaling to 1. In this study, two hidden layers are designed, and the softmax operator (Equation (16)) is added after the output of the second hidden layer. The function of softmax in this network is to scale the influence of hidden neurons by exponent arithmetic. The neuron with a larger weight will be enlarged after softmax operator. The neuron with a lower weight will be suppressed after softmax operator. Through this process, irrelevant weight is minimized from hidden neurons while useful weight is retained to make a greater contribution on output.

$$\begin{aligned} H &= Wx + b \\ \hat{y} &= softmax(H) \qquad \hat{y}_j = \frac{\exp(h_j)}{\sum_k \exp(h_k)} \end{aligned} \tag{16}$$

where W is the weights of the neural network, x is the input, b is the bias, h is the output of the second hidden layer, and exp is the exponential function.

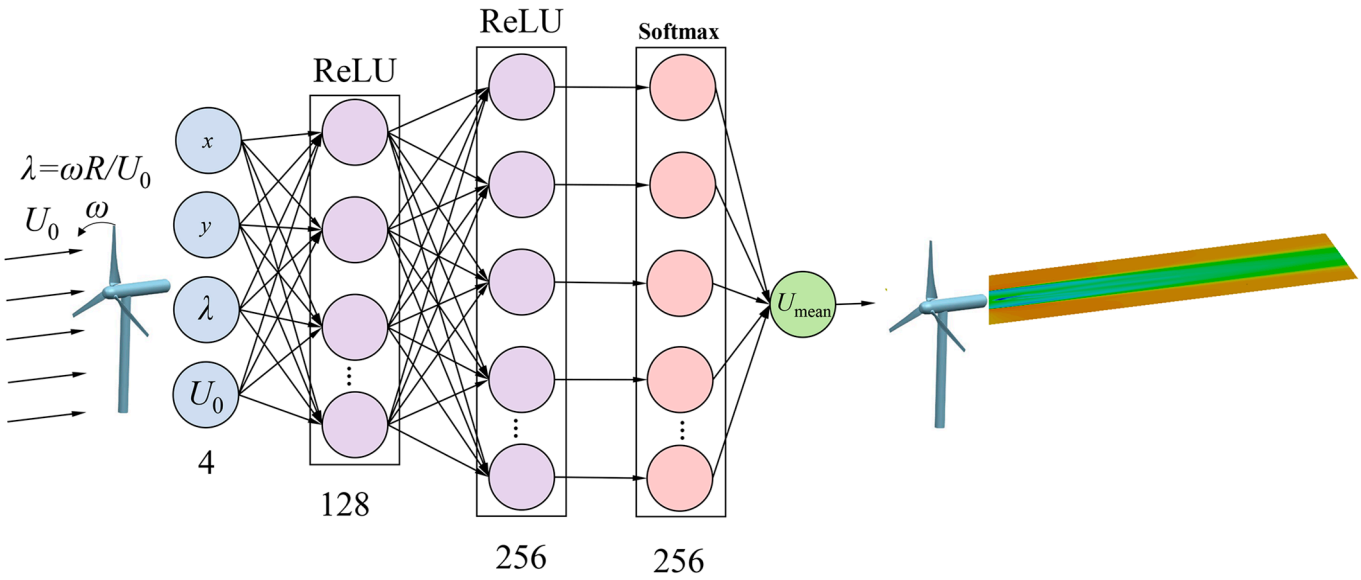


Figure 14. The architecture of a fully connected network with softmax layer.

4.4. Fully Connected Network with Residual Block

A fully connected network with a residual block (FCN-RB) is designed as shown in Figure 15. Analysis of FCN and FCN-C results indicates that large errors predominantly occur in the blade tip region, characterized by low velocity magnitudes and steep gradients. Therefore, these samples have a slight effect on backpropagation to update weights, especially in the earlier layers. The residual block (Equation (17)) is added to the FCN-C architecture as shown in Figure 15. The FCN-RB retains the original fully connected layers and adds a residual block to facilitate enough parameters and effective backpropagation.

$$y = \sigma(Wx + x) \tag{17}$$

where σ is the activation function, and x is the input of the neural network.

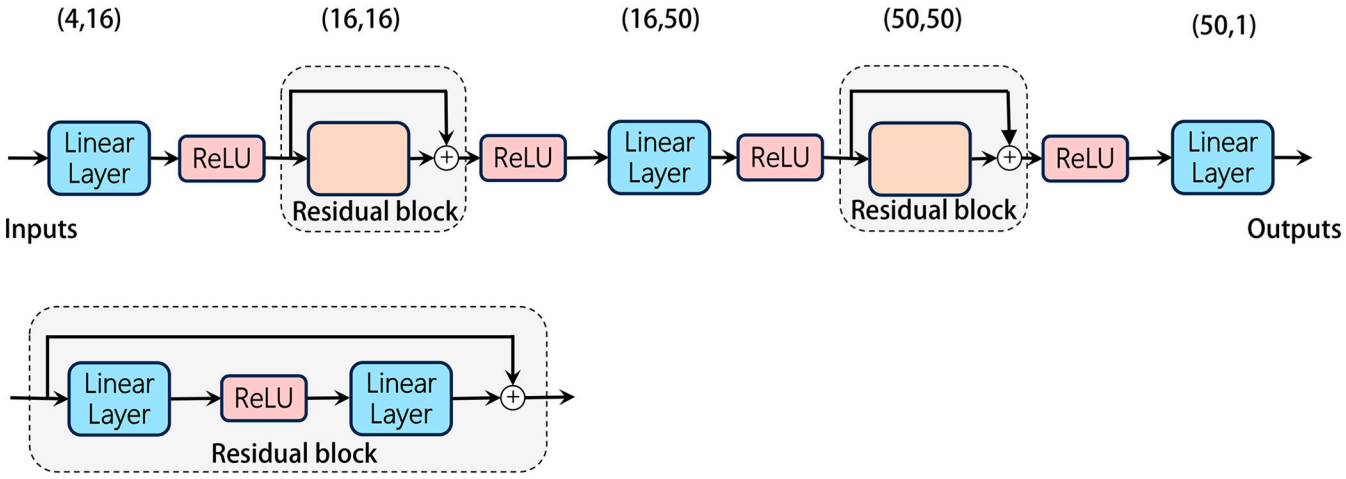


Figure 15. The architecture of a fully connected network with a residual block.

5. Results and Discussion of the Advanced Neural Network

The predictive results of pre-training and fine-tuning networks based on error classification with training data classification (PFNEC-TDC), Euclidean distance classification (PFNEC-EDC), and neural network classification (PFNEC-NNC) are shown in Figure 16. The results of FCN-S and FCN-RB are shown in Figure 17. Compared with FCN and FCN-C, the PFNEC improves a lot in prediction accuracy. As for the three classification models of PFNEC, EDC performs the best and achieves credible prediction accuracy for both external conditions. The NNC has a large deviation for high velocity performance, as shown in Figure 16. As for EDC, the classification is conducted based on Euclidean distance, and this mathematical information restricts the classification deviation on validation samples within a reasonable range. The classification of NNC is unrestricted mathematically. The prediction results of FCN-S and FCN-RB are slightly better than PFNEC for both external conditions, as shown in Figure 17. The FCN-S performs the best for prediction of (λ, U_0) = (7.3, 11.5) with R^2 = 0.9921 and 98.27% of the n = 76,230 data points have relative error within 10% ($-10\% \leq \delta \leq 10\%$) and 92.36% the of n = 76,230 data points have relative error within 5% ($-5\% \leq \delta \leq 5\%$). The FCN-RB performs the best for prediction of (λ, U_0) = (4.0, 8.5) with R^2 = 0.9891 and 99.69% the of n = 76,230 data points have relative error within 10% ($-10\% \leq \delta \leq 10\%$) and 96.26% the of n = 76,230 data points have relative error within 5% ($-5\% \leq \delta \leq 5\%$). The prediction results of PFNEC-EDC are close to the results of the optimal model. All models show good performance for the prediction of wind turbine wake flow, and note that the initialization of the neural network model during training has a non-negligible effect on prediction accuracy. Overall, the operating condition (λ, U_0) = (4.0, 8.5) is easier to learn and to infer than the (λ, U_0) = (7.3, 11.5) condition.

The contour of different models and their distribution of relative error are shown in Figures 18 and 19. It can be found that the high error region is mainly distributed around the blade tip region for the (λ, U_0) = (7.3, 11.5) condition and around the nacelle wake region for the (λ, U_0) = (4.0, 8.5). The reason is that higher TSR induces larger velocity gradients near the blade tip region, and it is difficult to capture this feature. When compared with CFD results, the models achieve acceptable accuracy. As analyzed in previous analysis, the FCN-RB performs the best for (λ, U_0) = (4.0, 8.5) condition, and FCN-S results perform best for (λ, U_0) = (7.3, 11.5) condition. PFNEC-EDC also has reliable performance for the prediction of both external conditions.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|^2 \tag{18}$$

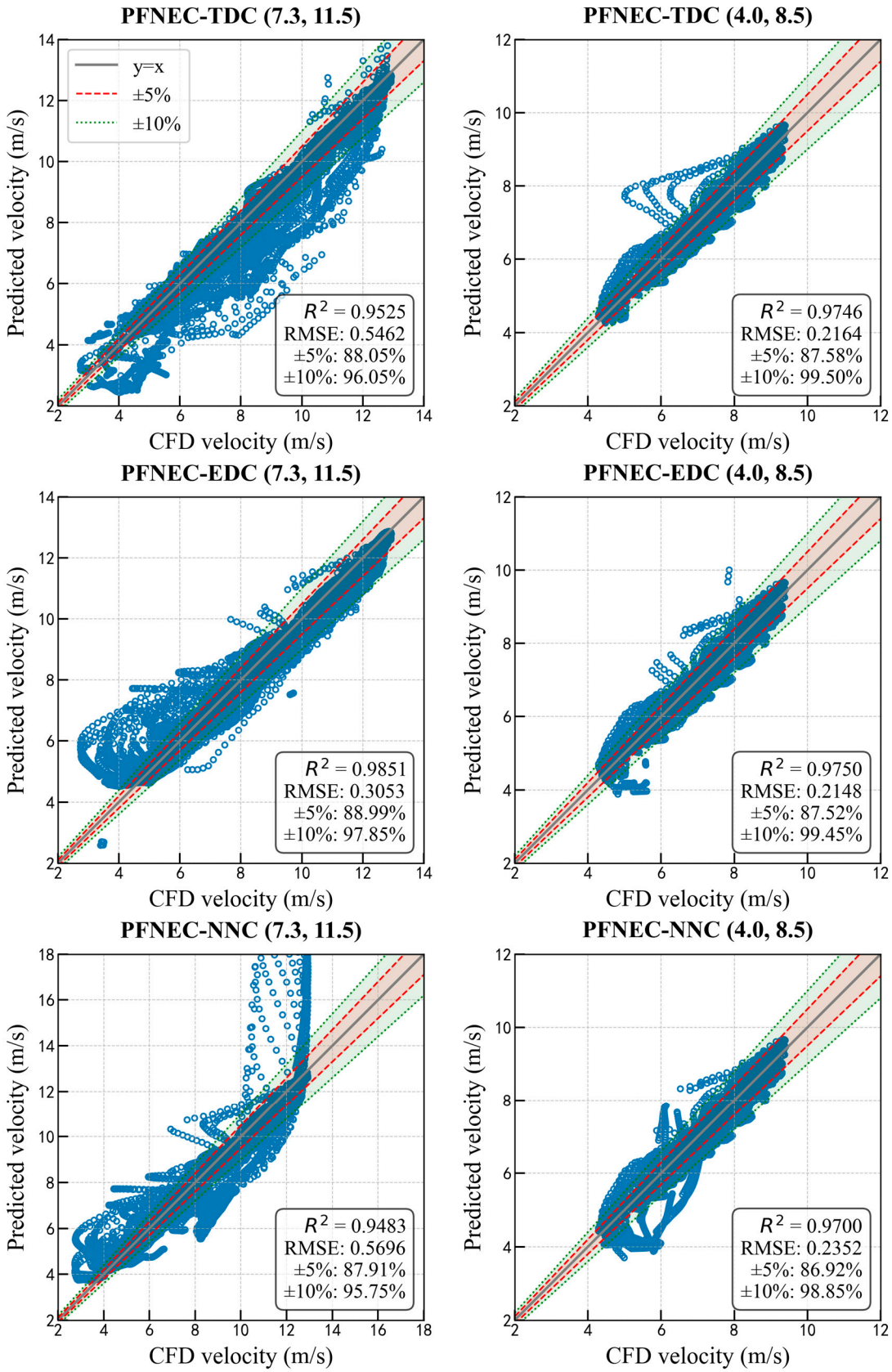


Figure 16. The distribution of mean velocity U_{mean} regressed by PFNEC-TDC, PFNEC-EDC, and PFNEC-NNC against CFD simulation results.

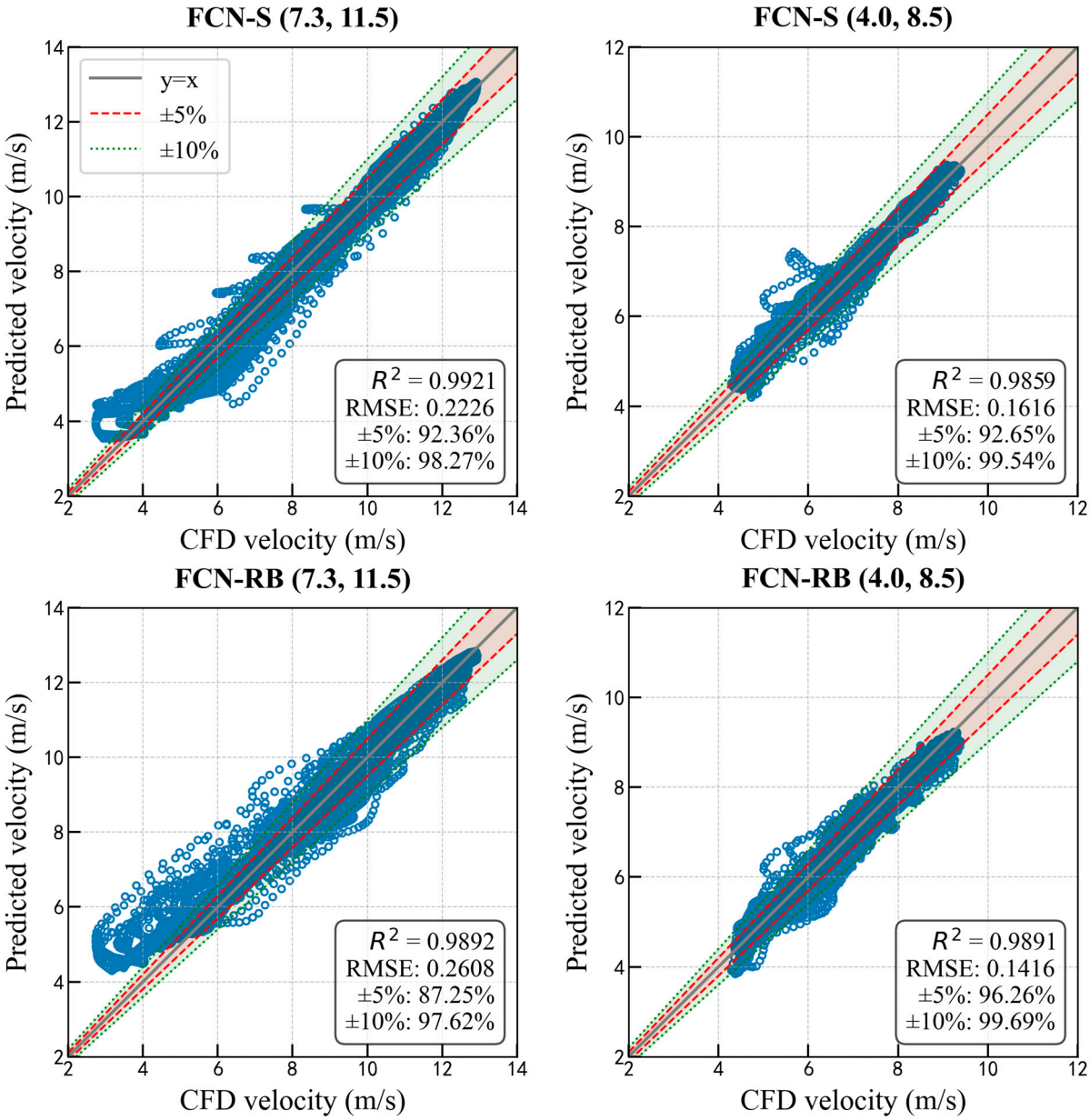


Figure 17. The distribution of mean velocity U_{mean} regressed by FCN-S and FCN-RB against CFD simulation results.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i| \tag{19}$$

$$MAPE = \frac{100}{N}\sum_{i=1}^{N}\frac{|\hat{y}_i - y_i|}{|y_i|} \tag{20}$$

$$sMAPE = \frac{100}{N}\sum_{i=1}^{N}\frac{2|\hat{y}_i - y_i|}{|\hat{y}_i + y_i|} \tag{21}$$

$$PAP = 100\left(1 - \frac{\sqrt{2}}{2}\sqrt{\left(\frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{y}_i - y_i|}{|y_i|}\right)^2 + \left(\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 / \sum_{i=1}^{N}(y_i - \overline{y})^2\right)^2}\right) \tag{22}$$

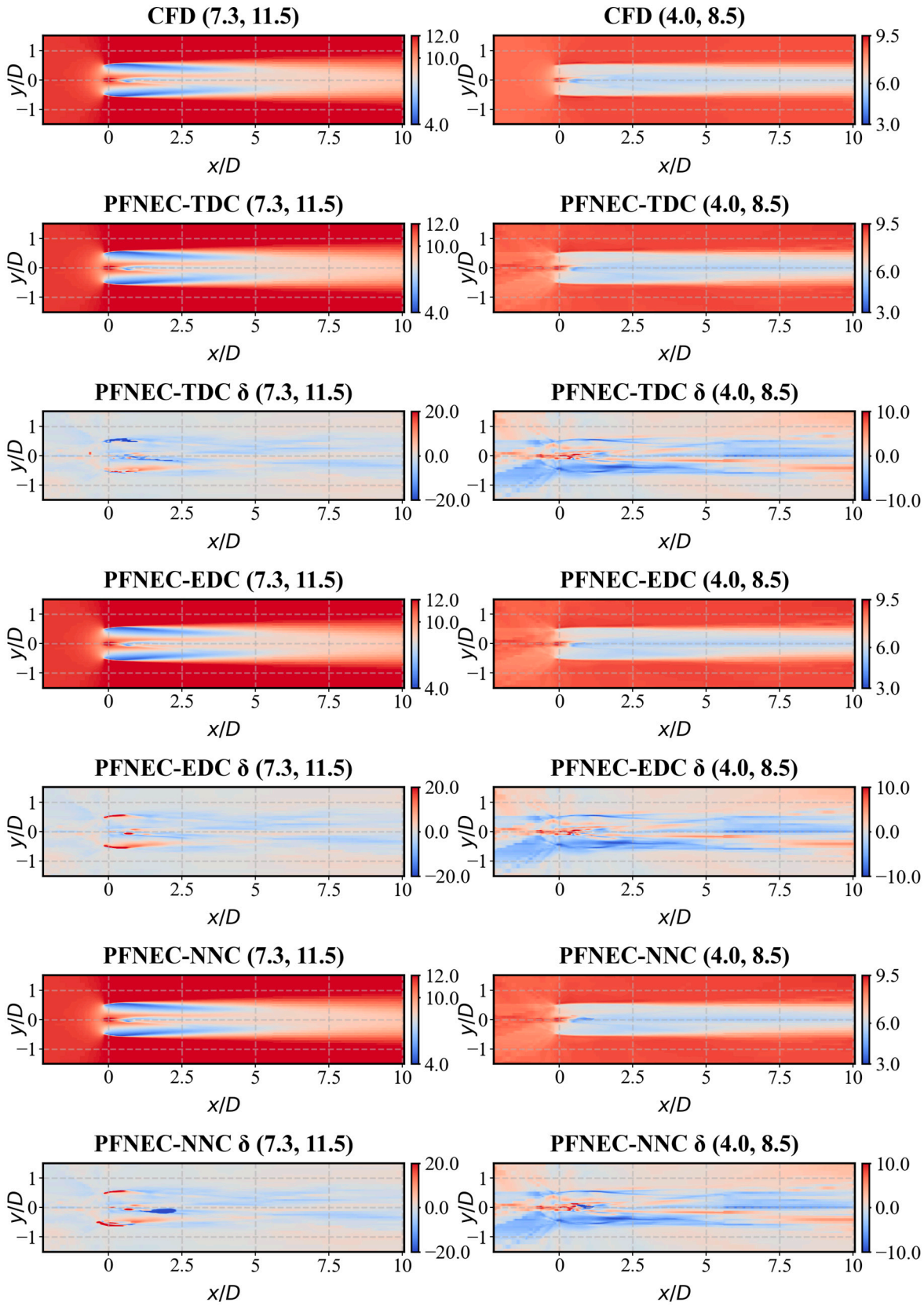


Figure 18. The comparison of PFNEC-TDC, PFNEC-EDC, and PFNEC-NNC with CFD simulation results of mean velocity U_{mean} on the XY plane.

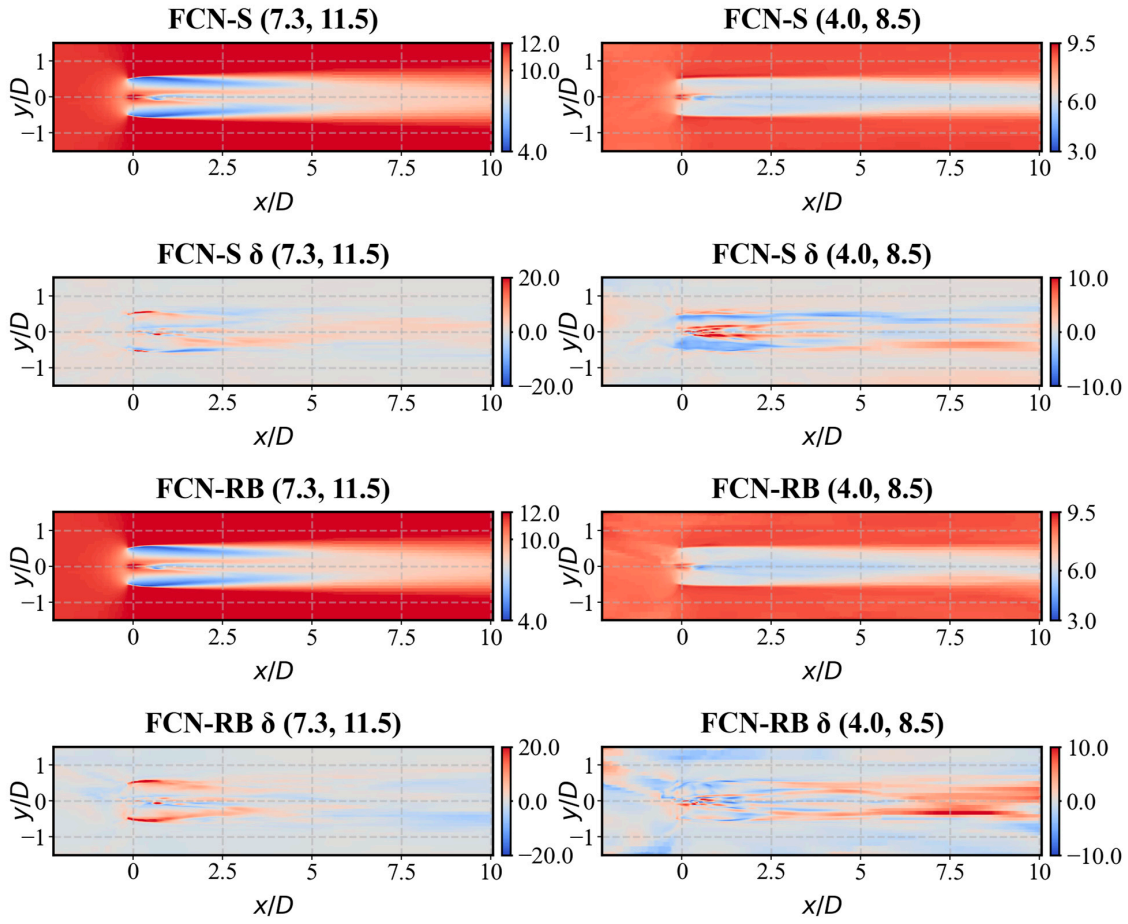


Figure 19. The comparison of FCN-S and FCN-RB with the CFD simulation results of mean velocity U_{mean} on the XY plane.

Six evaluation parameters are used to estimate the performance of different models. They include the coefficient of determination R^2 (Equation (13)), mean squared error (MSE) in Equation (18), mean absolute error (MAE) in Equation (19), mean absolute percentage error (MAPE) in Equation (20), symmetric mean absolute percentage error (sMAPE) in Equation (21), Percentage of Accuracy-Precision (PAP) [36] in Equation (22). The results of different models across six evaluation parameters are shown in Table 2. It is also evident that the FCN-RB performs best for the (λ, U_0) = (4.0, 8.5) condition, and the performance of FCN-S is the best for the (λ, U_0) = (7.3, 11.5) condition. The results of PFNEC-EDC are also acceptable and reliable.

Table 2. Performance comparison of different models.

Condition	Model	MSE	MAE	MAPE	sMAPE	R^2	PAP
(4.0, 8.5)	FCN	0.0914	0.2096	3.1233	3.2213	0.9505	96.8767
	FCN-C	0.0871	0.1988	3.0489	3.1216	0.9528	96.9511
	PFNEC-TDC	0.0468	0.1493	2.1437	2.1771	0.9746	97.8563
	PFNEC-EDC	0.0461	0.1494	2.1471	2.1845	0.9750	97.8529
	PFNEC-NNC	0.0553	0.1558	2.2598	2.3122	0.9700	97.7402
	FCN-S	0.0261	0.1076	1.6409	1.6462	0.9859	98.3591
	FCN-RB	**0.0200**	**0.0959**	**1.4041**	**1.4137**	**0.9891**	**98.5959**

Table 2. *Cont.*

Condition	Model	MSE	MAE	MAPE	sMAPE	R^2	PAP
(7.3, 11.5)	FCN	0.2879	0.3263	4.1991	4.0999	0.9541	95.8009
	FCN-C	0.1556	0.2529	3.3589	3.2704	0.9752	96.6411
	PFNEC-TDC	0.2983	0.2363	3.0491	3.2496	0.9525	96.9509
	PFNEC-EDC	0.0932	0.1950	2.4625	2.4193	0.9851	97.5375
	PFNEC-NNC	0.3244	0.2451	2.8887	2.8938	0.9483	97.1113
	FCN-S	**0.0495**	**0.1465**	**1.8542**	**1.8573**	**0.9921**	**98.1458**
	FCN-RB	0.0680	0.1768	2.3414	2.2746	0.9892	97.6586

6. Generalization of the Advanced Model with the Cross-Construction Method

The effectiveness of the cross-construction method has been validated by the performance of different neural networks in the previous section. It is necessary to estimate the performance of neural networks for wake prediction on extreme operating conditions and to test whether the present dataset can train a neural network that has the ability to predict wake flow under extreme operating conditions. Therefore, two conditions of (III (λ, U_0) = (3.0, 6.0) and IV (λ, U_0) = (10.2, 14.5)) located on the boundary of the dataset were selected to test the neural network trained on the present dataset, as shown in Figure 20.

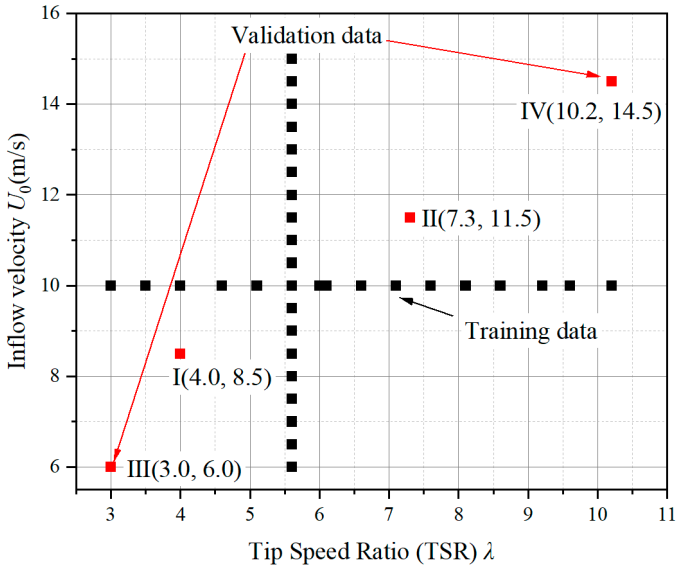


Figure 20. The distribution of validation data points in the cross-constructed dataset.

As concluded in the previous section, the FCN-S and FCN-RB perform the best for the wake prediction of Cases I and II. The predictions of the FCN-S and FCN-RB trained on the cross-construction dataset under validation Cases III and IV are shown in Figure 21. It is obvious that the large deviations are observed in the predictions for Cases III and IV compared with Cases I and II. As for the FCN-S model, the coefficient of determination R^2 = 0.4974 for case III (λ, U_0) = (3.0, 6.0) and R^2 = 0.7034 on IV (λ, U_0) = (10.2, 14.5) condition. The FCN-RB had the results of the coefficient of determination, R^2 = 0.7143 on III (λ, U_0) = (3.0, 6.0) condition and R^2 = 0.8448 on IV (λ, U_0) = (10.2, 14.5) condition. It is necessary to improve the prediction accuracy on extreme operating conditions while still retaining low computational cost.

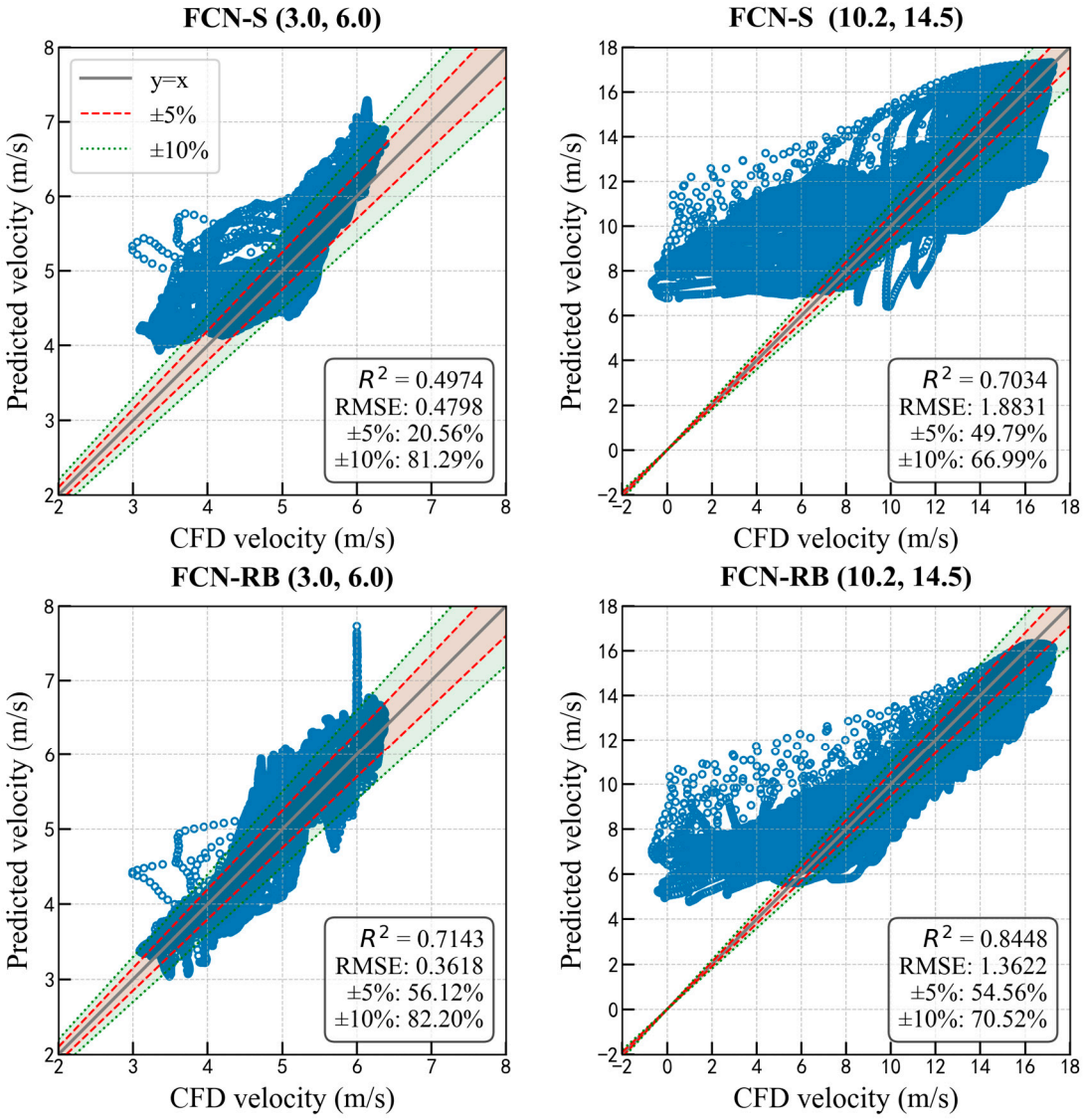


Figure 21. The distribution of mean velocity U_{mean} regressed by FCN-S and FCN-RB against CFD simulation results by the original dataset under III and IV conditions.

The flow features under extreme operating conditions should be included in the dataset to make the neural network learn these features. Four conditions located on the boundary of the dataset were added in modified cross-construction dataset, as shown in Figure 22. The four cases were marked with blue data points in Figure 22. The FCN-S and FCN-RB were retrained on a modified cross-construction dataset using the same initialization parameters. After training for 500 epochs, the prediction results under Cases III and IV are shown in Figure 23.

Figure 23 gives the scatter plot of prediction results compared with CFD results. It is obvious that the deviation decreases compared with the prediction results trained on the original dataset (as shown in Figure 8). The coefficient of determination is R^2 = 0.9545 for the FCN-S under case III (λ, U_0) = (3.0, 6.0) and R^2 = 0.8947 under case IV (λ, U_0) = (10.2, 14.5). The FCN-RB exhibits a further improvement in accuracy compared with the FCN-S, with R^2 = 0.9777 for case III (λ, U_0) = (3.0, 6.0) and R^2 = 0.9793 for case IV (λ, U_0) = (10.2, 14.5).

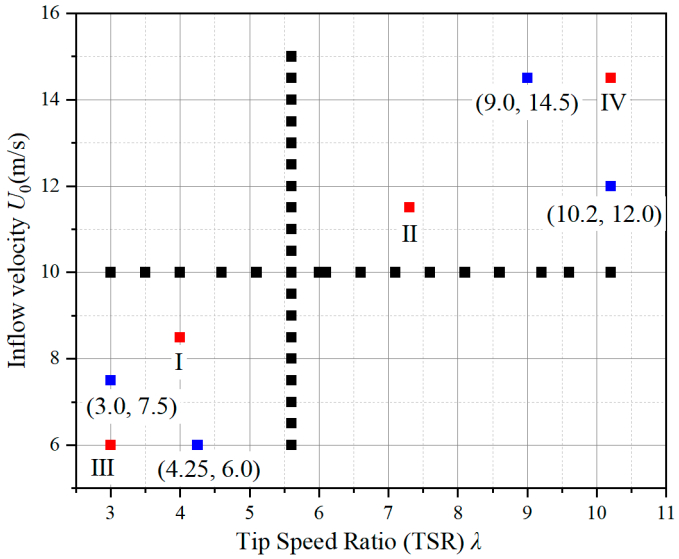


Figure 22. The distribution of data points in the modified dataset.

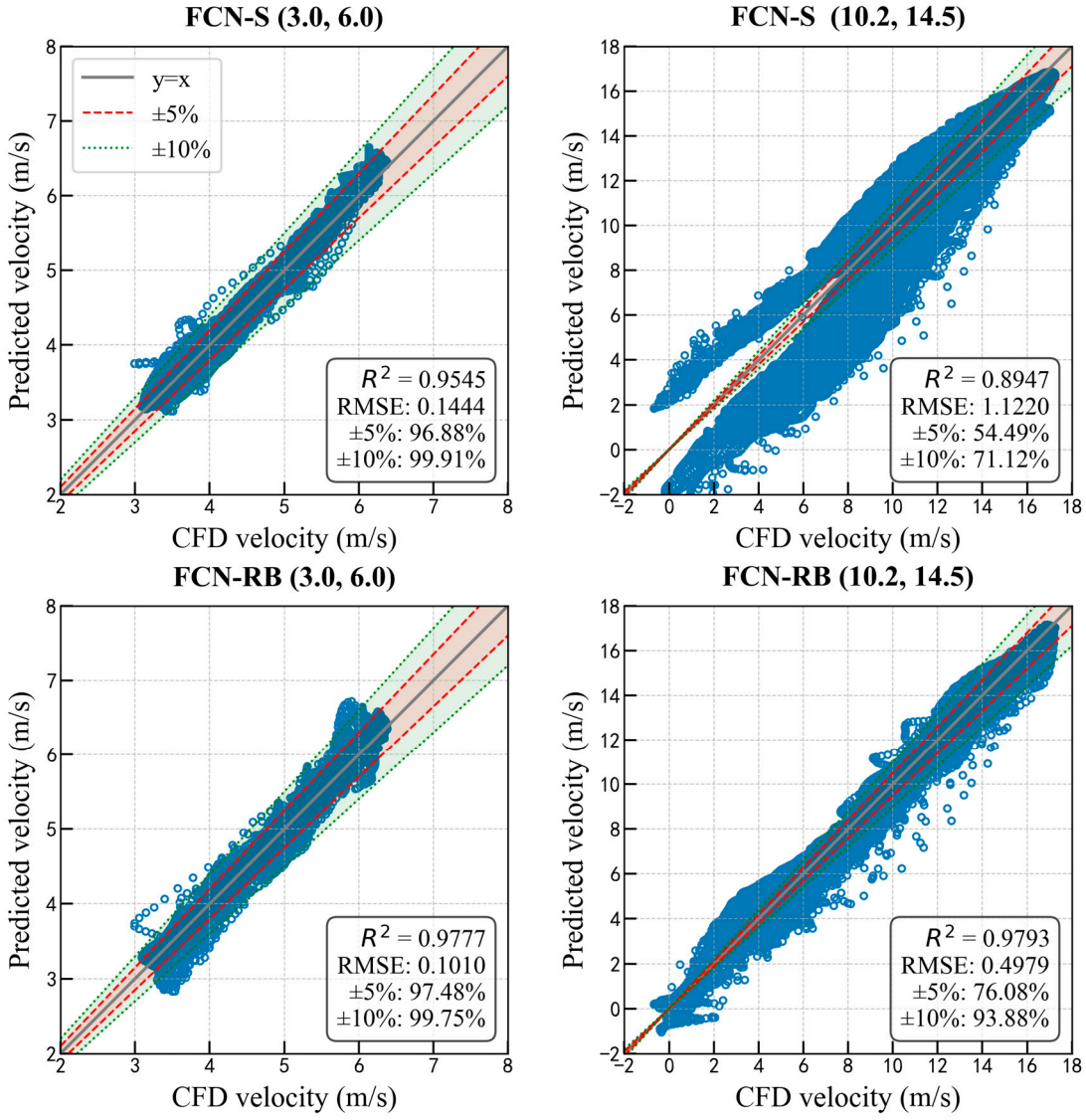


Figure 23. The distribution of mean velocity U_{mean} regressed by FCN-S and FCN-RB by the modified dataset against CFD simulation results under III and IV conditions.

By analyzing the characteristics of wind turbine wake and the distribution of predictive error, it is obvious that large errors occur near the blade tip wake region. At the blade tips, large velocity gradients occur, and blade tip vortices are induced so that the neural network cannot capture the gradient variation characteristics in this area. In order to enable the neural network to learn the complicated data containing multiple frequency components, a two-dimensional Fourier feature embedding was emploied [26]. The Fourier embedding is introduced as Equation (23):

$$\gamma(x,y) = \begin{bmatrix} \sin(2\pi x), \cos(2\pi x), \ldots, \sin\left(2^{i-1}\pi x\right), \cos\left(2^{i-1}\pi x\right), \\ \sin(2\pi y), \cos(2\pi y), \ldots, \sin\left(2^{j-1}\pi y\right), \cos\left(2^{j-1}\pi y\right) \end{bmatrix} \tag{23}$$

where γ is the Fourier operator, x and y are coordinate values.

Considering that the gradient in the y direction is larger than in the x direction, a higher embedding should be applied to y. Therefore, the i and j are set as:

$$\begin{aligned} i &= 1,2,3,4,5 \\ j &= 1,2,3,4,5,6,7,8,9,10 \end{aligned} \tag{24}$$

After Fourier embedding, the input features are extended to [$\gamma(x, y)$, x, y, λ, U_0]. The FCN-S with Fourier embedding (FCN-S-F) and FCN-RB with Fourier embedding (FCN-RB-F) were conducted with the original dataset and the modified cross-construction dataset. The prediction results trained on the original dataset (in Figure 8) are summarized in Table 3. The modified cross-construction dataset is summarized in Table 4.

Table 3. Performance comparison of different models with the original dataset.

Condition	Model	MSE	MAE	MAPE	sMAPE	R^2	PAP
(3.0, 6.0)	FCN-S	0.2302	0.4342	7.8150	7.6154	0.4974	92.1850
	FCN-RB	0.1309	0.2842	5.3668	5.2026	0.7143	94.6332
	FCN-S-F	0.0799	0.1744	3.6230	3.4378	0.8256	96.3770
	FCN-RB-F	**0.0585**	**0.1610**	**3.2685**	**3.1321**	**0.8723**	**96.7315**
(10.2, 14.5)	FCN-S	3.5460	1.1933	29.9165	12.4185	0.7034	70.0835
	FCN-RB	1.8557	**0.8942**	22.5147	9.9283	0.8448	77.4853
	FCN-S-F	**1.8426**	0.9375	**16.2334**	**9.5502**	**0.8459**	**83.7666**
	FCN-RB-F	2.7017	1.1286	26.3549	11.2388	0.7740	73.6451

Table 4. Performance comparison of different models with a modified dataset.

Condition	Model	MSE	MAE	MAPE	sMAPE	R^2	PAP
(3.0, 6.0)	FCN-S	0.0209	0.1277	2.2970	2.2717	0.9545	97.7030
	FCN-RB	0.0102	0.0699	1.3423	1.3402	0.9777	98.6577
	FCN-S-F	**0.0028**	**0.0399**	**0.7546**	**0.7506**	**0.9939**	**99.2454**
	FCN-RB-F	0.0030	0.0418	0.7936	0.7927	0.9933	99.2064
(10.2, 14.5)	FCN-S	1.2589	0.7698	12.8586	9.7683	0.8947	87.1414
	FCN-RB	0.2479	0.3582	4.4042	4.0426	0.9793	95.5958
	FCN-S-F	**0.1364**	**0.2402**	**2.6837**	**2.8291**	**0.9886**	**97.3163**
	FCN-RB-F	0.1888	0.2815	8.1298	3.6193	0.9842	91.8702

The results of FCN-S-F and FCN-RB-F in Tables 3 and 4 were obtained with different initialization schemes. It is evident in Table 3 that the prediction accuracy of FCN-S and FCN-RB was improved by employing Fourier embedding. This demonstrates that the Fourier embedding can enhance the generalization capacity of FCN-S and FCN-RB. Compared with the accuracy of FCN-S and FCN-RB for I (λ, U_0) = (4.0, 8.5) and II (λ, U_0) = (7.3, 11.5), the accuracy of FCN-S-F and FCN-RB-F for III (λ, U_0) = (3.0, 6.0) and IV (λ, U_0) = (10.2, 14.5) conditions remains relatively low. The results trained by the modified dataset of different neural networks are summarized in Table 4. Different networks were trained more effectively on the modified cross-construction dataset than on the original dataset. The FCN-RB performed better than the FCN-S in the prediction of extreme operating conditions. Therefore, the cross-construction method for dataset establishment is effective, and only sparse extreme operating conditions added in the cross-construction dataset can significantly enhance the model's generalization capability and its predictive performance under extreme operating conditions.

7. Conclusions

A cross-construction method was proposed to establish a sparse dataset for wind turbine wake, and the neural networks trained on this dataset achieved high accuracy and fidelity. The neural network models were designed in this study to make predictions of wind turbine wake with a cross-construction dataset. Some neural models were designed to improve the prediction accuracy of the wake flow field. The main conclusions are summarized as follows:

This study proposed a cross-construction method to generate a sparse yet representative CFD dataset for wind turbine wakes. The cross-construction method can build a sparse wind turbine wake dataset at a low cost, and the trained neural network has high prediction accuracy. The FCN-C predicted wake fields with the accuracy of R^2 = 0.9528, RMSE = 0.2952 under (λ, U_0) = (4.0, 8.5) condition and R^2 = 0.9752, RMSE = 0.3945 under the condition (λ, U_0) = (7.3, 11.5).

A pre-training and fine-tuning network based on error classification using three classification models was designed. The PFNEC improves the prediction accuracy for the wake flow field. PFNEC-EDC performed better than the other two models. PFNEC-EDC had the coefficient of determination R^2 = 0.9750 for (λ, U_0) = (4.0, 8.5) condition and R^2 = 0.9851 for (λ, U_0) = (7.3, 11.5).

Two models, FCN-S and FCN-RB, were developed and showed the best performance for the prediction of external conditions. The Fourier embedding can improve the predictive accuracy from R^2 = 0.7143 to R^2 = 0.8723 under (λ, U_0) = (3.0, 6.0) conditions and enhance the generalization capacity of neural networks. As for extreme operating conditions, adding four samples in the original dataset can improve the accuracy of extreme operating conditions from R^2 = 0.9545 to R^2 = 0.9939 under (λ, U_0) = (3.0, 6.0) condition and from R^2 = 0.8947 to R^2 = 0.9886 under (λ, U_0) = (10.2, 14.5) condition.

Limitations of this study should be noted. The cross-construction method was proposed and verified on a two-parameter space (TSR and inflow velocity), and its scalability to higher-dimensional conditions remains to be explored further. It is worthwhile to validate this method in a multi-parameter space.

Author Contributions: Conceptualization, Y.W. and Q.T.; methodology, Y.W. and Q.T.; software, Q.T., C.L., P.D. and J.W.; validation, Y.W. and Q.T.; formal analysis, Q.T.; investigation, Y.W. and Q.T.; resources, Y.W.; data curation, Q.T.; writing—original draft preparation, Q.T.; writing—review and editing, Y.W. and Q.T.; visualization, Q.T.; supervision, Y.W. and J.L.; project administration, Y.W.; funding acquisition, Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Fundamental Research Funds for the Central Universities (2022ZFJH004), the Creative Seed Fund of Shanxi Research Institute for Clean Energy, the Carbon Neutrality and Energy System Transformation project, and the National Science and Technology Major Project (2019-I-0022-0021).

Data Availability Statement: The data that support the findings of this study are available within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. GWEC. *Global Wind Report 2024*; Global Wind Energy Council: Brussels, Belgium, 2024.
2. Abdallah, A.; William, M.A.; Moharram, N.A.; Zidane, I.F. Boosting H-Darrieus vertical axis wind turbine performance: A CFD investigation of J-Blade aerodynamics. *Results Eng.* **2025**, *27*, 106358. [CrossRef]
3. Porté-Agel, F.; Bastankhah, M.; Shamsoddin, S. Wind-Turbine and Wind-Farm Flows: A Review. *Bound.-Layer Meteorol.* **2020**, *174*, 1–59. [CrossRef] [PubMed]
4. Jensen, N.O. *A Note on Wind Generator Interaction*; Risø National Laboratory: Roskilde, Denmark, 1983.
5. Bastankhah, M.; Porté-Agel, F. A new analytical model for wind-turbine wakes. *Renew. Energy* **2014**, *70*, 116–123. [CrossRef]
6. Zhang, H.; Gao, X.; Lu, H.; Zhao, Q.; Zhu, X.; Yu, W.; Fei, Z. Investigation of a new 3D wake model of offshore floating wind turbines subjected to the coupling effects of wind and wave. *Appl. Energy* **2024**, *365*, 123189. [CrossRef]
7. Brunton, S.L.; Noack, B.R.; Koumoutsakos, P. Machine Learning for Fluid Mechanics. *Annu. Rev. Fluid Mech.* **2020**, *52*, 477–508. [CrossRef]
8. Ye, M.; Li, M.; Liu, M.; Xiao, C.; Wan, D. Overview of Data-Driven Models for Wind Turbine Wake Flows. *J. Mar. Sci. Appl.* **2025**, *24*, 1–20. [CrossRef]
9. Wilson, B.; Wakes, S.; Mayo, M. Surrogate modeling a computational fluid dynamics-based wind turbine wake simulation using machine learning. In Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence (SSCI), Honolulu, HI, USA, 27 November–1 December 2017; pp. 1–8.
10. Purohit, S.; Ng, E.Y.K.; Kabir, I.F.S.A. Evaluation of three potential machine learning algorithms for predicting the velocity and turbulence intensity of a wind turbine wake. *Renew. Energy* **2022**, *184*, 405–420. [CrossRef]
11. Ti, Z.; Deng, X.W.; Yang, H. Wake modeling of wind turbines using machine learning. *Appl. Energy* **2020**, *257*, 114025. [CrossRef]
12. Ti, Z.; Deng, X.W.; Zhang, M. Artificial Neural Networks based wake model for power prediction of wind farm. *Renew. Energy* **2021**, *172*, 618–631. [CrossRef]
13. Nakhchi, M.E.; Naung, S.W.; Rahmati, M. Wake and power prediction of horizontal-axis wind farm under yaw-controlled conditions with machine learning. *Energy Convers. Manag.* **2023**, *296*, 117708. [CrossRef]
14. Renganathan, S.A.; Maulik, R.; Letizia, S.; Iungo, G.V. Data-driven wind turbine wake modeling via probabilistic machine learning. *Neural Comput. Appl.* **2022**, *34*, 6171–6186. [CrossRef]
15. Pujari, K.N.; Miriyala, S.S.; Mitra, K. Jensen-ANN: A Machine Learning adaptation of Jensen Wake Model. *IFAC-PapersOnLine* **2023**, *56*, 4651–4656. [CrossRef]
16. Liu, X.; Li, Z.; Yang, X. PhyWakeNet: A dynamic wake model accounting for aerodynamic force oscillations. *Wind Energy Sci. Discuss.* **2025**, *2025*, 1–29. [CrossRef]
17. Li, B.; Ge, M.; Li, X.; Liu, Y. A physics-guided machine learning framework for real-time dynamic wake prediction of wind turbines. *Phys. Fluids* **2024**, *36*, 035143. [CrossRef]
18. Zhang, J.; Zhao, X. Wind farm wake modeling based on deep convolutional conditional generative adversarial network. *Energy* **2022**, *238*, 121747. [CrossRef]
19. Zhang, Z.; Santoni, C.; Herges, T.; Sotiropoulos, F.; Khosronejad, A. Time-Averaged Wind Turbine Wake Flow Field Prediction Using Autoencoder Convolutional Neural Networks. *Energies* **2022**, *15*, 41. [CrossRef]
20. Schøler, J.P.; Rasmussen, F.P.W.; Quick, J.; Réthoré, P.E. Graph Neural Operator for windfarm wake flow. *Wind Energy Sci. Discuss.* **2025**, *2025*, 1–38.
21. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
22. Zhang, J.; Zhao, X. Digital twin of wind farms via physics-informed deep learning. *Energy Convers. Manag.* **2023**, *293*, 117507. [CrossRef]
23. Ctp, A.G.; Boya, S.K.; Jinka, R.; Gupta, A.; Tyagi, A.; Sarkar, S.; Subramani, D.N. A physics-informed neural network for turbulent wake simulations behind wind turbines. *Phys. Fluids* **2025**, *37*, 015110. [CrossRef]
24. Wang, L.; Dong, M.; Wang, L.; Huang, C.; Song, D.; Fan, X.; Yang, J.; Wang, T.; Chen, S.; Li, Q.A. Multi-scale wake modeling based on physics-informed neural networks and transfer learning. *Appl. Energy* **2026**, *406*, 127318. [CrossRef]

25. Kirby, A.; Briol, F.-X.; Dunstan, T.D.; Nishino, T. Data-driven modelling of turbine wake interactions and flow resistance in large wind farms. *Wind Energy* **2023**, *26*, 968–984. [CrossRef]
26. Wang, S.; Sankaran, S.; Perdikaris, P. Respecting causality for training physics-informed neural networks. *Comput. Methods Appl. Mech. Eng.* **2024**, *421*, 116813. [CrossRef]
27. Sedaghatizadeh, N.; Arjomandi, M.; Kelso, R.; Cazzolato, B.; Ghayesh, M.H. Modelling of wind turbine wake using large eddy simulation. *Renew. Energy* **2018**, *115*, 1166–1176. [CrossRef]
28. Ji, R.; Sun, K.; Zhang, J.; Zhu, R.; Wang, S. A novel actuator line-immersed boundary (AL-IB) hybrid approach for wake characteristics prediction of a horizontal-axis wind turbine. *Energy Convers. Manag.* **2022**, *253*, 115193. [CrossRef]
29. Zhao, M.; Chen, S.; Wang, K.; Wu, X.; Zha, R. Effect of the yaw angle on the aerodynamics of two tandem wind turbines by considering a dual-rotor wind turbine in front. *Ocean Eng.* **2023**, *283*, 114974. [CrossRef]
30. Kim, W.-W.; Menon, S. A new dynamic one-equation subgrid-scale model for large eddy simulations. In *Proceedings of the 33rd Aerospace Sciences Meeting and Exhibit, Reno, NV, USA, 9-12 January 1995*; American Institute of Aeronautics and Astronautics: Reston, VA, USA, 1995.
31. Huang, S.; Li, Q.S. A new dynamic one-equation subgrid-scale model for large eddy simulations. *Int. J. Numer. Methods Eng.* **2010**, *81*, 835–865. [CrossRef]
32. Sørensen, J.N.; Shen, W.Z. Numerical modeling of wind turbine wakes. *J. Fluids Eng.* **2002**, *124*, 393–399. [CrossRef]
33. Krogstad, P.-Å.; Eriksen, P.E. "Blind test" calculations of the performance and wake development for a model wind turbine. *Renew. Energy* **2013**, *a 50*, 325–333. [CrossRef]
34. Krogstad, P.Å.; Lund, J.A. An experimental and numerical study of the performance of a model turbine. *Wind Energy* **2012**, *15*, 443–457. [CrossRef]
35. Bengio, Y.; Louradour, J.; Collobert, R.; Weston, J. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, Montreal, QC, Canada, 14-18 June 2009*; Association for Computing Machinery: Montreal, QC, Canada, 2009; pp. 41–48.
36. Heidaryan, E. A Note on Model Selection Based on the Percentage of Accuracy-Precision. *J. Energy Resour. Technol.* **2018**, *141*, 045501. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.