Ensemble Offshore Wind Turbine Power Curve Modelling – An Integration of Isolation Forest, Fast Radial Basis Function Neural Network, and Metaheuristic Algorithm

Tenghui Li[a], Xiaolei Liu[a1], Zi Lin[b] and Rory Morrison[a]

[a] *James Watt School of Engineering, University of Glasgow, Glasgow, G12 8QQ, UK*

[b] *Department of Mechanical & Construction Engineering, Northumbria University, Newcastle upon Tyne, NE1 8ST, UK*

Abstract

Offshore wind energy is drawing increased attention for the decarbonization of electricity generation. Due to the unpredictable and complex nature of offshore aero-hydro dynamics, the Wind Turbine Power Curve (WTPC) model is an important tool for power forecasting and, hence, providing a reliable, predictable, and stable power supply. With the development of data-driven approaches, the Artificial Neural Network (ANN) has become a popular method for estimating WTPCs. This paper integrates the Isolation Forest (iForest), Nonsymmetric Fuzzy Means (NSFM) Radial Basis Neural Network (RBFNN), and metaheuristic algorithm to form a novel WTPC model. iForest performed anomaly detection and removal, NSFM RBFNN approximated the WTPC, and the metaheuristic solved NSFM optimization without training RBFNN. Four real-world datasets were used to assess the performance of NSFM RBFNN. According to multiple evaluation metrics and the Diebold-Mariano test, the accuracy of NSFM RBFNN was significantly better than the other competitive neural network-based methods. Additionally, NSFM RBFNN was shown to be more robust to anomalies than competitors, which is highly beneficial for practical applications.

Keywords: Offshore Wind Power; Wind Turbine Power Curve (WTPC); Radial Basis Function Neural Network (RBFNN).

Nomenclature			
Latin symbols		σ_l	width of l^{th} kernel
a	proportion coefficient	φ_l	output of l^{th} kernel
b	output offset	Λ	membership
c	average path length	Θ	cost of a possible solution
h	path length		

[1] Corresponding author, E-mail: Xiaolei.Liu@glasgow.ac.uk (XL)

$\mathbf{i}$	measurement instance	Abbreviation	
$\mathbf{L_{tb}}$	tabu list	ADAM	Adaptive Moment Estimation
P	probability of accepting a non-improving solution	ANFIS	Adaptive Neuro-Fuzzy Inference System
$\mathbf{p}$	fuzzy partition vector	ANN	Artificial Neural Network
R^2	coefficient of determination	DLNN	Deep Learning Neural Network
r	cooling rate	FM	Fuzzy Means
s	anomaly score	iForest	Isolation Forest
$\mathbf{s}_l$	l^{th} fuzzy subspace	MAE	Median Absolute Error
T_i	current temperature	MLPNN	Multilayer Perceptron Neural Network
w_l	synaptic weight of l^{th} kernel	MSE	Mean Square Error
$\mathbf{x}$	input vector	NRMSE	Normalized Root Mean Square Error
y	output response	NSFM	Nonsymmetric Fuzzy Means
y_n	n^{th} measured output	RBFNN	Radial Basis Function Neural Network
$\hat{y}_n$	n^{th} model output	SA	Simulated Annealing
$\bar{y}$	mean of measured outputs	SATS	Simulated Annealing Tabu Search
		SCADA	Supervisory Control and Data Acquisition
Greek symbols		SFM	Symmetric Fuzzy Means
γ_l	coefficient of l^{th} kernel	TS	Tabu Search
ε	relative distance	WT	Wind Turbine
$\boldsymbol{\mu}_l$	centre vector of l^{th} kernel	WTPC	Wind Turbine Power Curve

1. Introduction

With the increasing penetration of wind energy, power systems are taking the risk of unreliability and instability due to the stochastic nature of the wind resource being harnessed. Wind turbines (WT) based offshore are subject to nonlinear aero-hydro-dynamic environments, which results in difficulties in predicting the precise generated power [1]. As a consequence, appropriate approaches for estimating the power output of offshore WTs are critical for the energy industry for successful grid integration. Furthermore, accurate power estimations will be needed for optimal operation of hybrid technology power plants, such as with storage [2].

As a convenient way to derive the performance of a WT, a manufacturer will provide a Wind Turbine Power Curve (WTPC). The WTPC explicitly provides the output power of a WT for a range of wind speeds under specific test conditions. As the manufacturer's tests were conducted under ideal wind conditions, the WTPC will appear quite smooth. This is not typically the reality when a WT is deployed on-site as factors such as topography, roughness, and wake effects come into play. Furthermore, fluctuating wind speeds, directions, temperatures, and pressures will all serve to scatter output around the WTPC, as will endogenous factors such as blade condition [1, 3]. With the above challenges noted, WTPC modelling is still one of the key tools for monitoring the performance of WTs and forecasting their power output, and therefore developing accurate WTPC models can further benefit the application of wind power [3].

Techniques to fit the WTPC from sample data can be classified into parametric and nonparametric methods [1, 3]. A parametric model derives the output response through constructing a set of mathematical equations including a few parameters, which may be based on underlying physical laws [3]. However, owing to the uncertain nature of some parameters, it becomes more complex and more difficult to use parametric methods for deployed WTs. With floating offshore WTs in mind, parametric models become even harder to solve as additional degrees-of-freedom of the hub are added. In contrast to parametric methods, nonparametric methods do not require any prior assumptions to derive the relationship between input variables and output response, which provide higher reliability and lower estimation errors [1].

To monitor WTs and predict power output, many nonparametric methods of WTPC modelling have been developed and applied. For example, neural networks learned the relationship between input and output with the aid of nonlinear functions within neurons, data clustering methods characterized power curves by grouping similar data into classes or clusters, data mining methods extracted or mined the implicit information from mass data, and copula models considered the joint probability distribution of the output power and wind speed [3]. Nonparametric methods exhibit the suitability to establish models from large datasets, where the Artificial Neural Network (ANN) and Adaptive Neuro-Fuzzy Inference System (ANFIS) outperformed parametric and other nonparametric models [3]. ANN was claimed to be the best nonparametric model comparing with the Fuzzy C-Means clustering and data mining [4, 5].

As one of the most powerful nonparametric methods, ANNs have had many applications in WTPC modelling [6]. Li et al. designed a three-layer (4-8-1) Multilayer Perceptron Neural Network (MLPNN) with the compressing functions to estimate the output power from two sets of meteorological data with different sampling rates [7]. Mabel and Fernandez implemented a feed-forward three-layer (3-4-1) MLPNN to forecast the monthly power generation [8]. Pelletier et al. developed a multi-stage ANN modelling technique with six input features, where ANN was proven to be suitable for WTPC modelling and was able to reduce the absolute and random errors among parametric, nonparametric and discrete methods [9]. Jyothi and Rao introduced a method of one-step-ahead wind power forecasting based on an adaptive Wavelet Neural Network (WNN), which surpassed standard ANN and ANFIS methods [10]. Current research regarding ANN-based WTPC modelling is focused on optimizing architectures, using novel structures, guiding applications, and evaluating performance which contribute to precise prediction and convenient implementation. With the development of data-driven methods, many hybrid ANNs have been designed to deliver more accurate WTPC models. Shetty et al. developed a Radial Basis Function Neural Network (RBFNN) speeded up by an Extreme Learning Machine (ELM) and cooperating with the Particle Swarm Optimization Fuzzy C-Means (PSO-FCM) to obtain the power response [11]. Zhao et al. proposed a wind power forecasting system consisting of a numerical weather prediction (delivering real-time weather data series) and a three-layer feedforward ANN (cascaded with a power aggregation), in which wind speed data was processed by a Kalman filter and fed into a trained ANN with the rest of the weather data [12]. Liu et al. combined three individual forecasting models and an ANFIS model, where the Backpropagation Neural Network (BPNN), RBFNN, and Least Squares Support Vector Machine (LS-SVM) were included [13]. Liu et al. compared two anomaly detection techniques and integrated the Isolation Forest (iForest) with a five-layer Deep Learning Neural Network (DLNN) to forecast the offshore WT power [1]. Karamichailidou et al. applied a Nonsymmetrical Fuzzy Means (NSFM) RBFNN with the thin plate spline function optimized by the Tabu Search (TS) to establish WTPC models [14]. All mentioned approaches with their input features, sampling rate, and accuracy are summarized in **Table 1**. Compared to single ANNs, hybrid ANNs aim to improve the prediction accuracy, speed up the training process, and provide suggestions for data processing and hyperparameter optimization. The majority of current approaches were trained and tested with preprocessed data but the degradation with inaccurate measurements was not discussed for real-world implementation. Besides, the significance of prediction of a model with other competitive models was not assessed for offshore WTs.

Table 1
Comparison of WTPC models.

Reference	Training algorithm	Input features	Sampling rate	Accuracy
Li et al. [7]	MLPNN	wind speed, wind direction	5 sec, 10 min	Maximum Relative Error = 0.303-4.06%
Mabel and Fernandez [8]	MLPNN	wind speed, relative humidity, and generation hour	1 month	Mean Square Error (MSE) = 0.0065, Mean Absolute Error = 0.0586
Pelletier et al. [9]	MLPNN	wind speed, wind direction, yaw error, air density, turbulence intensity, and wind shear	1 sec	Weighted Mean Error = 0, Weighted Mean Absolute Error = 15.3-15.9
Jyothi and Rao [10]	WNN	wind speed, wind direction, wind density, and ambient temperature	10 min	Normalized Root Mean Square Error (NRMSE) = 0.02
Shetty et al. [11]	RBFNN, ELM, PSO-FCM	wind speed, wind direction, blade pitch angle, density, and rotor speed	1 hour	MSE = 3.004×10^{-4}, Mean Relative Error = 2.948%
Zhao et al. [12]	Numerical weather prediction, MLPNN,	wind speed, wind direction, temperature, pressure, and	15 min	NRMSE = 16.47%

	Kalman filter	humidity		
Liu et al. [13]	BPNN, RBFNN, LS-SVM, ANFIS	wind speed, wind direction, and temperature	15 min	Mean Absolute Percentage Error = 6.70-11.76, Normalized Mean Absolute Error = 1.01-4.23, NRMSE = 2.37-6.24
Liu et al. [1]	iForest, MLPNN	wind speed, nacelle orientation, yaw error, blade pitch angle, and ambient temperature	1 sec	MSE = 0.003258
Karamichailidou et al. [14]	RBFNN, NSFM, TS	wind speed, wind direction, blade pitch angle, and ambient temperature	10 min	Median Absolute Error = 19.3407-23.6915

Most studies concerned a select number of meteorological features for their WTPC modelling, typically concentrated on wind speed, wind direction, and factors related to air density. Wind speed and wind direction relative to the orientation of the WT, known as yaw, have a large influence on wind power. Considering the rapidly fluctuating nature of wind fields, it is practically impossible for a WT to be aligned with the incoming wind, or unyawed, at any time. For this paper, wind direction, and hence yaw angle, is not available so nacelle orientation is used instead. The blade pitch angle is related to the aerodynamic efficiency that affects the output power by varying the angle of attack [14]. The air density is mainly affected by the ambient temperature, whose importance cannot be neglected [14]. Therefore, input attributes in this study are the wind speed, nacelle orientation, blade pitch angle, and ambient temperature. The output response is the WT active power.

The main contributions of this paper to fill the current knowledge gap are described below:

a. Most of the current ANN-based WTPC models have been designed, tested, and compared using onshore WTs. This paper uses data from an offshore wind farm. Four WT datasets, including a case of scattered outputs, are used to assess the performance, in which the scattered case is caused by the grid stability, maintenance, safety, etc.
b. Although NSFM RBFNN has been previously applied to onshore WTPC modelling [14], this paper proposes a different nonsymmetric partition technique and a novel kernel function for offshore wind power. Additionally, a novel metaheuristic algorithm, based on the Simulated Annealing (SA) and improved with a memorable tabu list to avoid duplicate solution explorations, is introduced to solve the optimal nonsymmetric fuzzy partition for NSFM, in which an objective function regarding relative distance instead of evaluation metrics of RBFNN is adopted to avoid training RBFNN.
c. Many previous WTPC methods only considered numerical metrics, such as the MSE, to evaluate and compare the accuracy. This paper relies on multiple metrics but also carries out a significance test, the

Diebold-Mariano test, on two model outputs produced by the proposed method and any one of the competitive methods. The model degradation under inaccurate measurements is also studied to investigate the feasibility of applying the proposed method to real-world WTs.

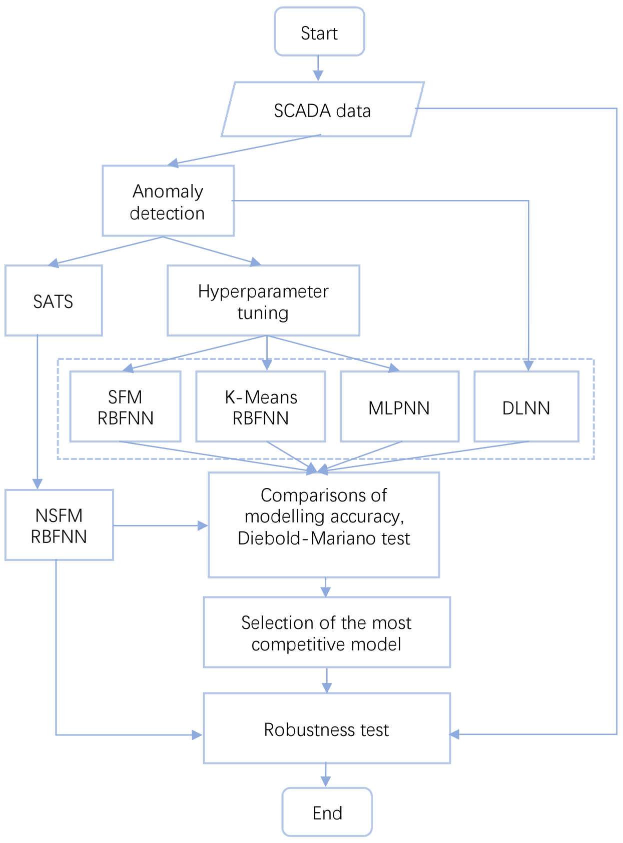


Fig. 1. Flowchart of investigating WTPC modelling.

As shown in the flowchart in **Fig. 1**, the remainder of this paper is organized as follows: **Section 2** provides characteristics of the target WTs and details of the datasets. **Section 3** introduces our proposed RBFNN model which receives data processed by an anomaly detection technique. To initialize RBF kernels, a revised NSFM approach was developed, which induced the motivation of applying a novel multivariate RBF kernel function. This was optimized by a hybrid metaheuristic method, the Simulated-Annealing-Tabu-Search (SATS). **Section 4** presents the results of the anomaly detection and evaluates the performance of NSFM RBFNN by comparing against Symmetric Fuzzy Means (SFM) RBFNN, K-Means RBFNN, MLPNN, and DLNN. This section also includes the application of the Diebold-Mariano test, which is applied to evaluate the significance of the accuracy of model forecasting and select the most competitive method. At the end of the comparison, a robustness test is carried out on the NSFM RBFNN and the other selected method to investigate the models' accuracy under raw measurements including missing data, sampling errors, and noise. **Section 5** summarizes the main findings of the presented paper.

2. Data description

Four Supervisory Control and Data Acquisition (SCADA) datasets were used for training and testing, which were measured by four Siemens SWT-3.6-120 3.6 MW offshore WTs from the same wind farm at Anholt. Anholt is Denmark’ s largest offshore wind farm, it was constructed and is still managed by the company Ørsted. The layout of Anholt and four selected WTs (introduced in **Section 2.2**) are presented in **Fig. 2**, in which the wind farm is approximately 20 km long and up to 5 km wide.

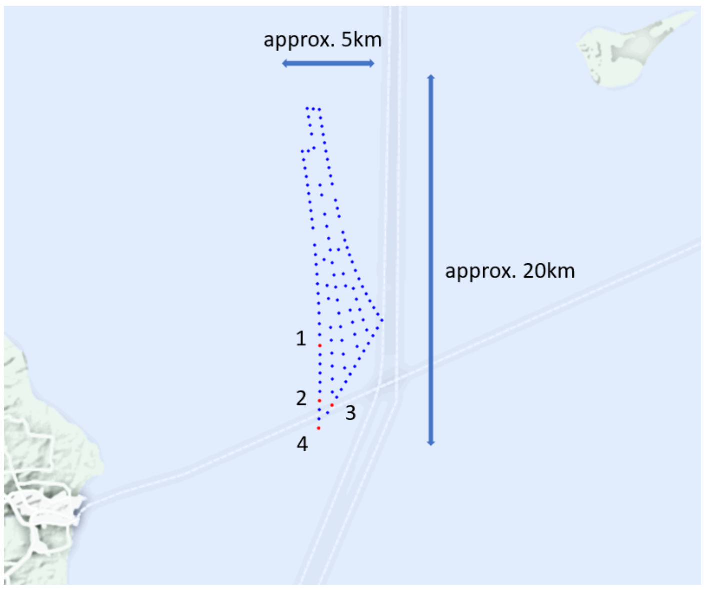


Fig. 2. Layout of Anholt wind farm.

2.1 Target wind turbines

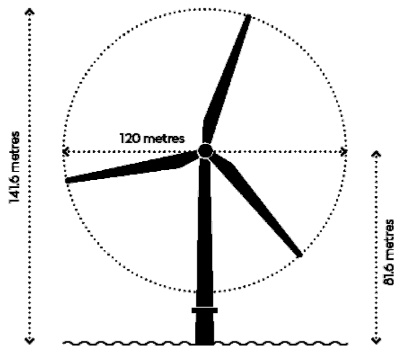


Fig. 3. Dimensions of the WTs of Anholt wind farm.

All WTs are identical 3-bladed horizontal axis WTs, fixed to the seabed via monopiles. The WTs have rotor diameters of 120 m, hub heights of 81.6 m, and tip heights of 141.6 m, as illustrated in **Fig. 3**. For operation and control, the cut-in and nominal power wind speeds are designed with the ranges of 3-5 and 12-13 m/s, respectively [15]. The WTs are variable speed and pitch regulated machines. The main shaft rotation speed can be tuned within the range of 5-13 rpm. An asynchronous generator couples with a gearbox and a mechanical brake. Other specifications of the target WTs are described in **Table 2** [15].

Table 2

Technical key data of WTs.

Properties	Value
Capacity	3.6 MW
Nominal Rotor speed	5~13 rpm
Rotor tilt	6 deg
Hub height	81.6 m
Rotor diameter	120 m
Blade length	58.5 m
Gearbox ratio	1:119
Yaw type	active
Cut-in wind speed	3~5 m/s
Nominal power wind speed	12~13 m/s
Cut-out wind speed	25 m/s

2.2 SCADA data description

The SCADA data were measured from January 2013 to December 2014, the power curves are shown in **Fig. 4**. The data was provided at 10-minute intervals. The datasets were reduced to only the parameters described in **Section 1**, which are: wind speed, nacelle orientation, blade pitch angle, ambient temperature, and active power output. The raw SCADA data contained abnormal measurements that were considered out of reasonable boundaries. To filter out these, we only accept data with measurements of wind speed between 0 and 40 m/s, nacelle orientation between -720 and 720°, pitch angle between -2 and 80°, ambient temperature between -15 and 35 C°, and active power between 0 and 3603 kW. It is noted that a valid nacelle orientation outside ±360° will be modulo of -360° or 360° for the negative or positive angle to eliminate the round angle. Additionally, if one or more parameters of an instance were missing the entire instance was deleted.

The power generation of each WT was influenced by weather conditions and external constraints, which led to different power outputs at the same time, although these WTs were at a neighbour location. As shown in

Fig. 4, each WT had a relatively unique dispersion of the growth between zero and the rated output, which led to that WT-4 was the widest (as a scattered case), WT-3 was slightly wider than WT-1 and 2. The highlighted measurements for WT-3 and WT-4 are also called type ii anomalies, this will be introduced in **Section 3.1**.

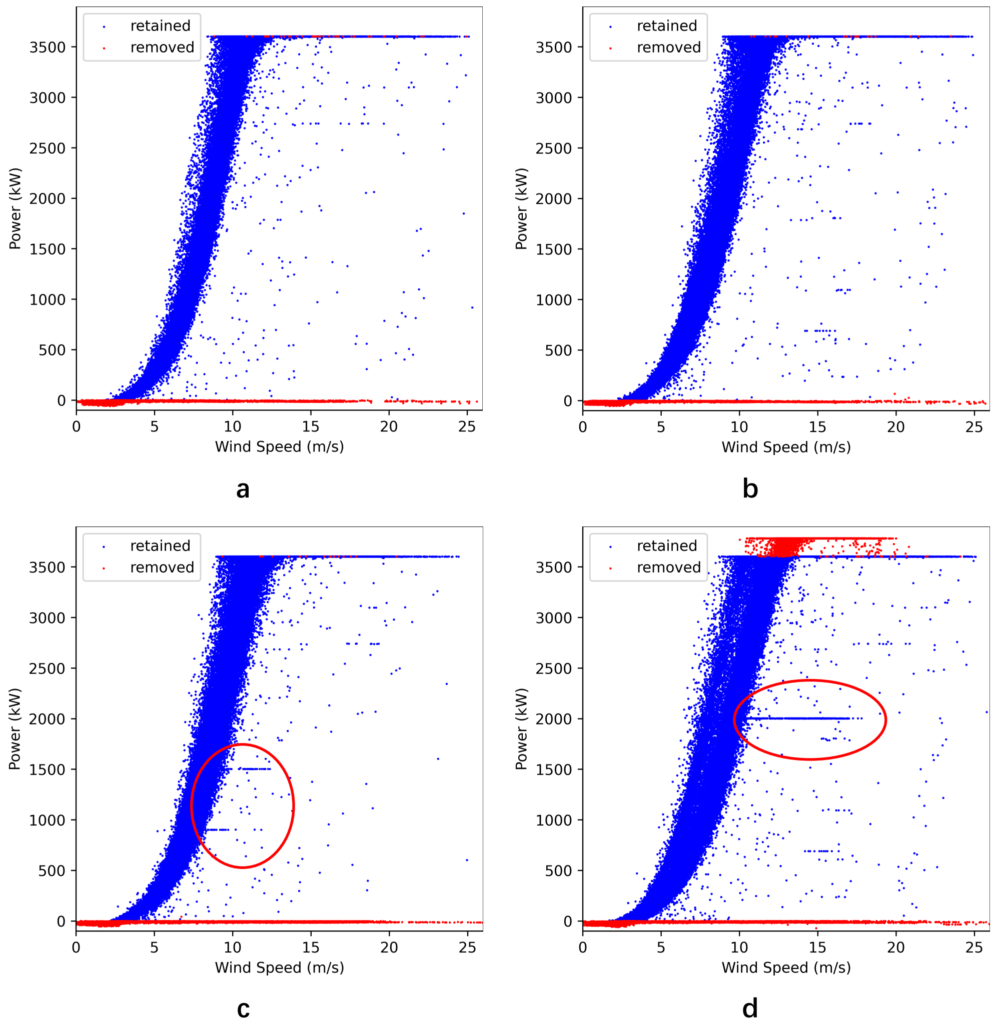


Fig. 4. Power curves: (a), (b), (c), and (d) for WT-1, 2, 3, and 4, respectively. Abnormal measurements, which are considered unreasonable, have been removed. However, anomalies clearly remain, as highlighted.

The statistical details, including count, mean, standard deviation, and percentiles, are presented in **Table 3**. Each SCADA set contains 105,120 rows, corresponding to 10-minute data for 2 years. After removing obvious anomalies, the number of instances in each dataset is not the same, which is 93,392, 93,043, 92,182, and 86,929, for WT-1, 2, 3, and 4 respectively. As observed, most type i anomalies are detected and removed by applying our data-filtering criteria. The lower number of available data of WT-4 is mainly due to large amounts of missing and incorrect measurements. The datasets share similar statistical characteristics in wind speed, pitch angle, and temperature. However, the differences in nacelle orientation cannot be ignored since these are the outcomes of yaw control. This implies that the underlying physical models are likely not identical and, as such, individual modelling of the WTs is necessary.

Table 3

Statistics of SCADA for the 4 WTs.

WT	Statistic	Wind speed, m/s	Nacelle orientation, °	Blade pitch angle, °	Ambient temperature, ℃	Active power, kW
1	Count	93392	93392	93392	93392	93392
	Mean	9.17	118.19	2.76	2071.81	8.91
	Standard deviation	3.67	145.87	5.86	1390.48	6.36
	Minimum	1.92	-359.29	-1.40	0.01	-6.00
	25%	6.34	27.28	-0.91	653.88	4.00
	Median	8.76	142.13	-0.46	2123.67	9.00
	75%	11.71	232.31	6.12	3601.32	14.00
	Maximum	25.32	359.97	79.03	3603.00	26.97
2	Count	93043	93043	93043	93043	93043
	Mean	9.67	106.09	2.86	2077.21	9.09
	Standard deviation	3.82	152.59	5.90	1391.07	6.31
	Minimum	2.26	-359.94	-1.40	0.01	-6.00
	25%	6.67	6.46	-0.91	660.43	4.00
	Median	9.30	133.58	-0.47	2122.84	9.00
	75%	12.39	223.10	6.46	3601.41	14.00
	Maximum	24.91	359.98	79.18	3602.96	26.54
3	Count	92182	92182	92182	92182	92182
	Mean	9.60	97.45	2.82	2061.09	8.78
	Standard deviation	3.83	150.92	5.82	1392.40	6.43
	Minimum	2.26	-359.76	-1.40	0.01	-6.18
	25%	6.58	3.16	-0.91	638.52	4.00
	Median	9.16	122.45	-0.48	2084.32	9.00
	75%	12.42	213.84	6.39	3601.35	14.00
	Maximum	24.96	360.00	79.78	3603.00	27.00
4	Count	86929	86929	86929	86929	86929
	Mean	9.79	76.70	2.61	2037.69	9.45
	Standard deviation	4.02	156.20	5.89	1382.00	6.57
	Minimum	1.70	-359.81	-1.20	0.00	-5.63
	25%	6.60	-64.30	-0.91	632.43	4.07
	Median	9.40	99.54	-0.47	2016.59	10.00
	75%	12.60	205.75	5.47	3601.30	15.00
	Maximum	25.78	360.00	79.91	3603.00	27.23

3. Methodology

Our proposed method consists of 3 components: anomaly detection via iForest, an improved RBFNN integrating with a kernel initialization method for predicting power, and a novel hybrid metaheuristic for optimization. To train a model effectively and efficiently, a single anomaly detection method is used to prevent abnormal SCADA measurements from affecting the training process. The key of our offshore WTPC modelling is an RBFNN, which benefits from simple architecture and the capability of approximating nonlinear systems [16]. Though some previous studies explored RBFNNs for onshore WTPC modelling [11, 13, 14], this study focused on delivering RBFNN-based offshore WTPC model with higher accuracy, faster convergence, and greater robustness, which was achieved by introducing NSFM to initialize RBF kernels and a novel kernel function with multiple widths to provide a better approximation. Also, a metaheuristic was suggested to find a solution for NSFM optimization.

3.1 Anomaly detection

Anomalies are instances that appear to be generated by a different mechanism to that generating the rest of the data [17]. To generate an accurate WTPC model from real-world SCADA data, anomalies must be removed prior to training. For WT SCADA data, anomalies can be classified into the following three types (illustrated in **Fig. 5**):

Type i: No power output at wind speeds is significantly greater than the cut-in wind speed. This is WT stoppage, typically for maintenance or grid demands [18].

Type ii: Power is constrained and lower than the theoretically achievable value. This is WT curtailment, typically due to grid demands [19].

Type iii: Power is randomly distributed in the vicinity of the theoretical value. Measurement failures, sensor degradations, and sampling noises are responsible for this type, as well as transitions from type i/ ii to normal operation [20].

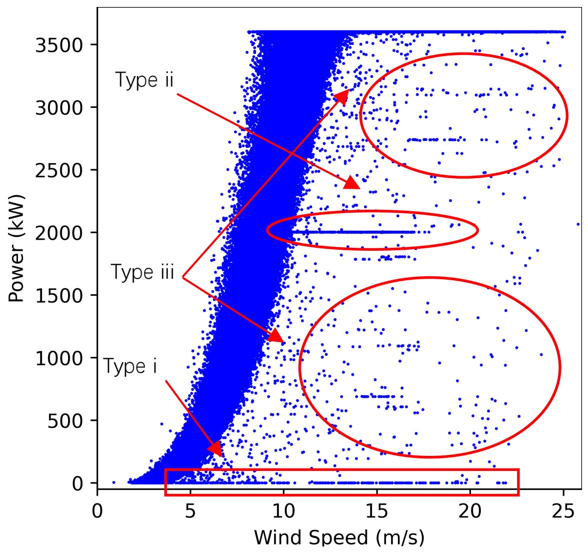


Fig. 5. Example of different type errors in a power curve.

To detect anomalies, this paper adopts iForest, a model-based approach, that makes use of two of their properties: they are a minority consisting of fewer instances, and they have attribute values that differ from those of normal instances [21]. The procedure of implementing iForest includes generating isolation trees after subsampling, calculating the path length $h(\mathbf{i})$ of an instance $\mathbf{i}$ (i.e. a measurement of input and output) according to isolation trees, and evaluating the anomaly score s of $\mathbf{i}$ by Eq. (1) to identify if $\mathbf{i}$ belongs to anomalies (it is considered as an anomaly when s is close to 1) [21].

$$s(\mathbf{i}, n) = 2^{-\frac{E(h(\mathbf{i}))}{c(n)}} \tag{1}$$

where $E(h(\mathbf{i}))$ is the average path length of $h(\mathbf{i})$ from a collection of isolation trees and $c(n)$ is the average path length of an unsuccessful search in a binary search tree.

3.2 Radial basis function neural network

This paper adopts a typical 3-layer RBFNN, as shown in **Fig. 6**, in which the single hidden layer consists of nonlinear RBF kernels and synaptic weights. The output φ_l of the l-th Gaussian kernel is given by Eq. (2) [22].

$$\varphi_l(\mathbf{x}) = e^{-\gamma_l \|\mathbf{x}-\boldsymbol{\mu}_l\|^2} \tag{2}$$

where $\mathbf{x}$ is an input vector (i.e. the n-th input vector $\mathbf{x}_n$ from the N input data $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n, \dots, \mathbf{x}_N]$, $\mathbf{x}$ replaces $\mathbf{x}_n$ in the following for simplicity), $\boldsymbol{\mu}_l$ is the centre vector of the l-th kernel, and γ_l is the coefficient of the l-th kernel, also written as $1/2\sigma_l^2$ in the form of the kernel width or radius σ_l.

The output response y of a given L kernels RBFNN is expressed by Eq. (3) [22].

$$\mathrm{y} = \sum_{l=1}^{L} w_l \varphi_l(\mathbf{x}) \tag{3}$$

where w_l is the synaptic weight of the l-th kernel. A variation of Eq. (3) is to include an offset b [22], as per Eq. (4). This is used in this paper to achieve a better approximation.

$$\mathrm{y} = \sum_{l=1}^{L} w_l \varphi_l(\mathbf{x}) + b \tag{4}$$

Before training traditional RBFNNs, the hyperparameter for the number of kernels, L, must be given to initialize the structure. For an L Gaussian kernels RBFNN, the following values need to be determined in the stage of training:

- RBF centres;

- RBF coefficients;
- synaptic weights and an offset.

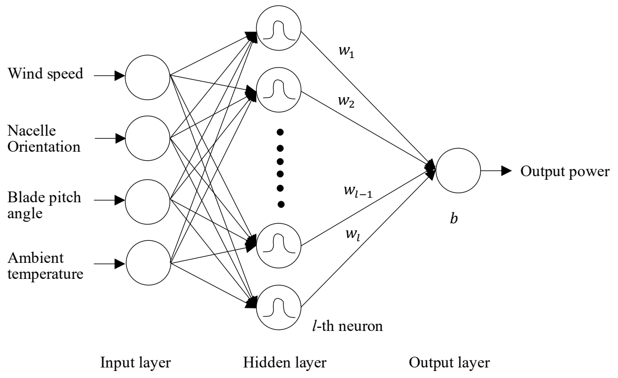


Fig. 6. Architecture of RBFNN for offshore WTPC modelling.

One popular algorithm determining RBF centres is based on the K-Means clustering, where an RBF centre is assumed to be at the centre of a cluster of the input space [23]. The main disadvantage of K-Means RBFNN is that the number of clusters has effects on the final estimation. On the other hand, the number of clusters cannot be automatically determined due to the intrinsic nature of K-Means. Therefore, another algorithm based on the Fuzzy Means (FM) is developed to obtain the number of RBF kernels and their centres and widths simultaneously [24].

3.3 Nonsymmetric fuzzy means

FM algorithms make use of the fuzzy partition technique to divide a multi-dimensional input space into a set of fuzzy subspaces, which results in the Symmetric Fuzzy Means and Nonsymmetric Fuzzy Means [24, 25]. NSFM divides each dimension of the input space into several evenly distributed sets without interfering with the other dimensions [25]. Accordingly, SFM is a special case of NSFM where the same partition acts on all dimensions [24]. Many studies have successfully applied NSFM in initializing RBFNN [14, 25-27]. The implementation of NSFM is adapted in this section for use of offshore WTs, which includes a nonsymmetric fuzzy partition and an NSFM algorithm.

Consider an input vector $\mathbf{x}$ with D features:

$$\mathbf{x} = [x_1, x_2, \dots, x_d, \dots, x_D] \tag{5}$$

where x_d is the value of the d-th dimension of $\mathbf{x}$.

The vector $\mathbf{p}$ can partition the input space into several nonsymmetric fuzzy subspaces.

$$\mathbf{p} = [p_1, p_2, \dots, p_d, \dots, p_D] \tag{6}$$

where p_d is the partition to an input variable x_d.

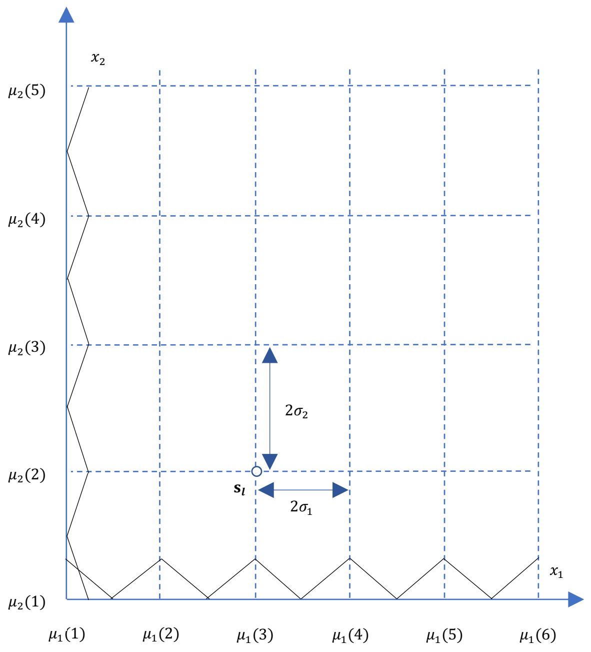


Fig. 7. Example of nonsymmetric fuzzy partition on a 2D input space.

For a given vector $\mathbf{p}$, the l-th selected fuzzy subspace $\mathbf{s}_l$ containing D triangular fuzzy sets is expressed in Eq. (7).

$$\begin{aligned}\mathbf{s}_l &= \left[s_{l,1}, s_{l,2}, \dots, s_{l,d}, \dots, s_{l,D}\right] \\ &= \begin{Bmatrix} \left[\mu_{l,1}(\eta_1), \sigma_{l,1}\right], \left[\mu_{l,2}(\eta_2), \sigma_{l,2}\right], \dots, \\ \left[\mu_{l,d}(\eta_d), \sigma_{l,d}\right], \dots, \left[\mu_{l,D}(\eta_D), \sigma_{l,D}\right] \end{Bmatrix}\end{aligned} \tag{7}$$

where η_d $(\in [1, p_d])$ is the selected partition number on the d-th dimension, $\mu_{l,d}(\eta_d)/\sigma_{l,d}$ is the centre/width of the selected d-th dimensional fuzzy set $s_{l,d}$. **Fig. 7** illustrates how a nonsymmetric fuzzy partition works on a 2D input space. In this case, $\mathbf{p} = [6,5]$ partitions x_1 and x_2 into 6 and 5 fuzzy sets with intervals σ_1 and σ_2. The selected fuzzy subspace $\mathbf{s}_l$ in **Fig. 7** is expressed as $\{[\mu_1(3), \sigma_1], [\mu_2(2), \sigma_2]\}$. It is noted that fuzzy sets of each dimension are not intersected, which is different from the previous studies [14, 25-27].

The NSFM algorithm aims to find a fuzzy space $\mathbf{S}$, which is to say a compact subset of fuzzy subspaces $[\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_l, \dots, \mathbf{s}_L]$, covering all input data rather than the whole input space under the condition of a given

$\mathbf{p}$ [25]. It is noted that the number of RBF kernels is the same as the number of selected fuzzy subspaces L, which is automatically determined by NSFM algorithm. Consequently, RBF centres are placed on the centres of selected fuzzy subspaces [25]. Using anomaly detection on the dataset, the fuzzy space can be shrunk to reduce the number of RBF kernels as anomalies tend to make NSFM import excessive fuzzy subspaces.

To calculate the membership of $\mathbf{x}$ regarding $\mathbf{s}_l$, the relative distance $\varepsilon_{\mathbf{s}_l}(\mathbf{x})$ between them is defined by Eq. (8) [25].

$$\varepsilon_{\mathbf{s}_l}(\mathbf{x}) = \sqrt{\frac{1}{D}\sum_{d=1}^{D}\left(\frac{x_d - \mu_{l,d}(\eta_d)}{\sigma_{l,d}}\right)^2} \qquad (8)$$

The multidimensional membership function $\Lambda_{\mathbf{s}_l}(\mathbf{x})$ is given by Eq. (9) [28].

$$\Lambda_{\mathbf{s}_l}(\mathbf{x}) = \begin{cases} 1 - \varepsilon_{\mathbf{s}_l}(\mathbf{x}), & \varepsilon_{\mathbf{s}_l}(\mathbf{x}) \leq 1 \\ 0, & \varepsilon_{\mathbf{s}_l}(\mathbf{x}) > 1 \end{cases} \qquad (9)$$

The relative distance $\varepsilon_{s_{l,d}}(x_d)$ of an input variable x_d to a selected fuzzy set $s_{l,d}$ can be expressed by Eq. (10).

$$\varepsilon_{s_{l,d}}(x_d) = \left|\frac{x_d - \mu_{l,d}(\eta_d)}{\sigma_{l,d}}\right| \qquad (10)$$

With the introduction of $\varepsilon_{s_{l,d}}(x_d)$, the progression of selecting a potential fuzzy set is more efficient because Eq. (10) makes the search process independent on each dimension.

The NSFM algorithm starts by identifying if the n-th input vector $\mathbf{x}_n$ is already in the current fuzzy space $\mathbf{S}$. If $\mathbf{x}_n$ is outside $\mathbf{S}$, which is implied by the fact that $\Lambda_{\mathbf{s}_l}(\mathbf{x}_n)$ regarding an arbitrary fuzzy subspace $\mathbf{s}_l$ of $\mathbf{S}$ is always zero (otherwise $\mathbf{x}_n$ is in $\mathbf{S}$ when there exists at least a nonzero membership), a new fuzzy subspace $\mathbf{s}_{L+1}$ will be searched to cover this input vector and added into $\mathbf{S}$. The principle of searching $\mathbf{s}_{L+1}$ is based on the minimum relative distance $\varepsilon_{s_{L+1,d}}(x_d)$. The NSFM algorithm will iterate through the rest of the input data $\mathbf{X}$ after initializing $\mathbf{S}$ so the number of iterations is equal to $N-1$. The search for fuzzy subspaces can be regarded as a kind of greedy strategy, so the sequence of input data may affect the final fuzzy space, especially when a SCADA dataset maintains its time series characteristic. To minimize this effect, it is recommended to randomize the order of an input time series.

To make full use of NSFM’ s strengths, a novel multivariate kernel function $\varphi_l(\mathbf{x})$ described in Eq. (11) is introduced here to replace Eq. (2) as the kernel function.

$$\varphi_l(\mathbf{x}) = e^{-\frac{1}{2}\sum_d^D\left(\frac{x_d-\mu_{l,d}}{\sigma_{l,d}}\right)^2} \tag{11}$$

Algorithm 1: NSFM algorithm

```
Input:   $\mathbf{X} = [\mathbf{x_1}, \mathbf{x_2}, \dots, \mathbf{x_n}, \dots, \mathbf{x_N}]$
         $\mathbf{p} = [p_1, p_2, \dots, p_d, \dots, p_D]$
Output:  $\mathbf{S} = [\mathbf{s_1}, \mathbf{s_2}, \dots, \mathbf{s_l}, \dots, \mathbf{s_L}]$
1.  initialize $\mathbf{S} = [\mathbf{s_1}]$ by taking $\mathbf{x_1}$ into steps (11-14)
2.  For $n = 2{:}N$ Do:
3.     obtain $L$ of $\mathbf{S}$
4.     For $l = 1{:}L$ Do:
5.        If $\mathbf{x_n}$ is covered by $\mathbf{s_l}$ according to Eq. (9)
6.           $\mathbf{x_n}$ is in $\mathbf{S}$ and steps (10-16) will be skipped
7.           break
8.        End If
9.     End For
10.    If $\mathbf{x_n}$ is outside $\mathbf{S}$
11.       For $d = 1{:}D$ Do:
12.          select $\mu_{L+1,d}(\eta_d)$ and $\sigma_{L+1,d}$ with the minimum $\varepsilon_{s_{L+1,d}}(x_d)$ calculated by Eq. (10)
13.       End For
14.          $\mathbf{s_{L+1}} = \left\{ \begin{matrix} [\mu_{L+1,1}(\eta_1), \sigma_{L+1,1}], \\ [\mu_{L+1,2}(\eta_2), \sigma_{L+1,2}], \\ \dots, \\ [\mu_{L+1,D}(\eta_D), \sigma_{L+1,D}] \end{matrix} \right\}$
15.       add $\mathbf{s_{L+1}}$ into $\mathbf{S}$ and update $L = L + 1$
16.    End If
17. End For
```

The traditional kernel function in Eq. (2) uses a single width to calculate, Eq. (11) adopts independent widths for all dimensions, which effectively deliver the information from input variables to a neuron. However, introducing this kernel function will not aggravate the computational burden since all widths have been obtained during the selection of the minimum $\varepsilon_{s_{L+1,d}}(x_d)$. Since the nonsymmetric fuzzy partition influences the performance of NSFM RBFNN, it is necessary to find the optimal solution of the nonsymmetric fuzzy partition, which can be regarded as an optimization problem. Many metaheuristics have been developed to solve this problem. The next section will describe a hybrid metaheuristic method to solve NSFM optimization.

3.4 Hybrid metaheuristic algorithm

Many metaheuristic methods have been developed to escape local optima in the search for global optima. Metaheuristics can be classified into population-based and single solution-based methods to explore the solution space. Although some [25, 29] applied population-based metaheuristics (Genetic Algorithm and Particle Swarm Optimization) to derive a solution of NSFM, these were found to be computationally intensive with huge memory consumptions and lengthy search times. Furthermore, the situation is exacerbated as datasets get larger. Therefore, a single solution-based metaheuristic is preferred for SCADA. Simulated Annealing is a stochastic, single solution-based metaheuristic, which always accepts a better solution and is capable of accepting a non-improving solution with a varied probability [27, 30, 31]. Tabu Search is also a single solution-based metaheuristic, which starts with a random solution and a memory-based tabu list [31]. In each iteration, the neighbourhood is fully explored to generate the local best solution. If the evaluation of this solution is improved, the tabu list will always be updated [30, 31]. Meanwhile, the tabu list also accepts a non-improving solution to avoid local optima [14, 30, 31]. However, basic SA may explore a solution multiple times, and searching the whole neighbourhood in TS is time-consuming. Therefore, a hybrid Simulated-Annealing-Tabu-Search (SATS) algorithm is developed to find a solution for the nonsymmetric fuzzy partition. The basic idea of SATS is to add a tabu list into SA to avoid duplicated searches.

SATS algorithm, **Algorithm 2**, starts with an initial solution $\mathbf{p}$ and a tabu list $\mathbf{L_{tb}}$. The solution space of $\mathbf{p}$ in this paper is set to the lower and upper bounds ($\mathbf{p_{min}} = [6,6,6,6]$ and $\mathbf{p_{max}} = [11,11,11,11]$). After that, a neighbour solution $\mathbf{p_{ngb}}$ in the vicinity of $\mathbf{p}$ within $\Delta\mathbf{p}$ is randomly selected, which is not contained by $\mathbf{L_{tb}}$ and will be added into $\mathbf{L_{tb}}$ to avoid repetitive explorations. If $\mathbf{L_{tb}}$ includes all neighbour solutions, the program will be terminated. Subsequently, $\mathbf{p_{ngb}}$ is evaluated by the cost function Θ in Eq. (12) after **Algorithm 1** is executed to obtain $\mathbf{S} = [\mathbf{s_1}, \mathbf{s_2}, \dots, \mathbf{s}_l, \dots, \mathbf{s}_L]$ under the condition of $\mathbf{p_{ngb}}$.

$$\Theta(\mathbf{p}) = \sum_{n=1}^{N} \max \begin{pmatrix} \Lambda_{\mathbf{s_1}}(\mathbf{x}_n), \Lambda_{\mathbf{s_2}}(\mathbf{x}_n), \\ \dots, \Lambda_{\mathbf{s}_L}(\mathbf{x}_n) \end{pmatrix} \qquad (12)$$

Algorithm 2: SATS algorithm

```
Input:   $\mathbf{X} = [\mathbf{x_1}, \mathbf{x_2}, \dots, \mathbf{x}_n, \dots, \mathbf{x}_N]$
         $\Delta\mathbf{p} = [\Delta p_1, \Delta p_2, \dots, \Delta p_d, \dots, \Delta p_D]$
         $\mathrm{T}_0,\ r$
Output:  $\mathbf{p_{opt}} = [p_{1,opt}, p_{2,opt}, \dots, p_{d,opt}, \dots, p_{D,opt}]$
1.   generate a random solution $\mathbf{p}$
2.   add $\mathbf{p}$ into $\mathbf{L_{tb}}$
3.   While not termination Repeat:
```

```
4.      generate a neighbourhood solution $\mathbf{p_{ngb}}$ not in $\mathbf{L_{tb}}$
5.      add $\mathbf{p_{ngb}}$ to $\mathbf{L_{tb}}$ and run **Algorithm 1**
6.      **If** $\Theta(\mathbf{p_{ngb}}) > \Theta(\mathbf{p})$
7.        accept $\mathbf{p} = \mathbf{p_{ngb}}$
8.      **Else**
9.        **If** $P_1 < P(\mathbf{p_{ngb}})$
10.         release $\mathbf{p}$ from $\mathbf{L_{tb}}$ and accept $\mathbf{p} = \mathbf{p_{ngb}}$
11.     **End If**
12.     update $\mathrm{T}_{i+1}$ by using Eq. (14)
13. **End While**
14. **Return** $\mathbf{p_{opt}}$ from $\mathbf{L_{tb}}$
```

The commonly used method for evaluating the performance of a hyperparameter optimization technique for ANNs is to compare a metric (such as MSE) on the testing dataset after training ANNs. To avoid the time-consuming training process, the cost function of Eq. (12) is defined to evaluate different nonsymmetric fuzzy partition solutions. With the help of the cost function, this metaheuristic does not need to train RBFNN. As such, it is a nontrainable parameter optimization technique, which can save the training time of RBFNN. This cost function calculates the membership of each input vector in a fuzzy space. An input vector may belong to multiple fuzzy subspaces, which leads to ambiguous membership determination, hence only the maximum membership among different fuzzy subspaces is considered. If $\mathbf{p_{ngb}}$ is believed as a better solution, $\mathbf{p}$ will always be updated. However, $\mathbf{p_{ngb}}$ can also be accepted when the probability condition, i.e. $P_1 < P(\mathbf{p_{ngb}})$, of accepting a non-improving solution is satisfied, where P_1 has the standard uniform distribution $P_1 \sim U(0,1)$ and $P(\mathbf{p_{ngb}})$ is calculated by Eq. (13). This is revised from Ref. [31].

$$P(\mathbf{p_{ngb}}) = a \cdot e^{-\frac{\Theta(\mathbf{p_{ngb}}) - \Theta(\mathbf{p})}{\mathrm{T}_i}} \qquad (13)$$

where a is the proportion coefficient and T_i is the current temperature. a is set to 0.5 in this paper.

Accepting the current non-improving solution is the so-called aspiration criterion in TS. The key step of our proposed SATS algorithm is to release $\mathbf{p}$ before updating so that the neighbourhood of $\mathbf{p}$ can be searched again. At the end of each iteration, the current temperature T_i is updated by Eq. (14) [31].

$$\mathrm{T}_{i+1} = r \cdot \mathrm{T}_i \qquad (14)$$

where r is the cooling rate. In this paper, the initial temperature and cooling rate are set to 10,000 and 0.98 respectively. It is noted that the initial temperature should be adjusted according to the quantity of data to make the annealing process smooth.

4. Case study

The methodology outlined in **Section 3** was applied to the SCADA data outlined in **Section 2**.

4.1 Result of isolation forest

iForest was implemented using the Scikit-Learn library. The number of trees, subsampling ratio, contamination ratio, and maximum features, were 200, 0.8, 0.3, and 4 respectively. **Fig. 8** displays the iForest-filtered results, in which the anomalies highlighted in **Fig. 4** have been removed. After filtering, these datasets were ready to train and test NSFM RBFNN. In this study, each filtered dataset is randomly divided into three groups, 70% for training, 20% for testing, and 10% for validation.

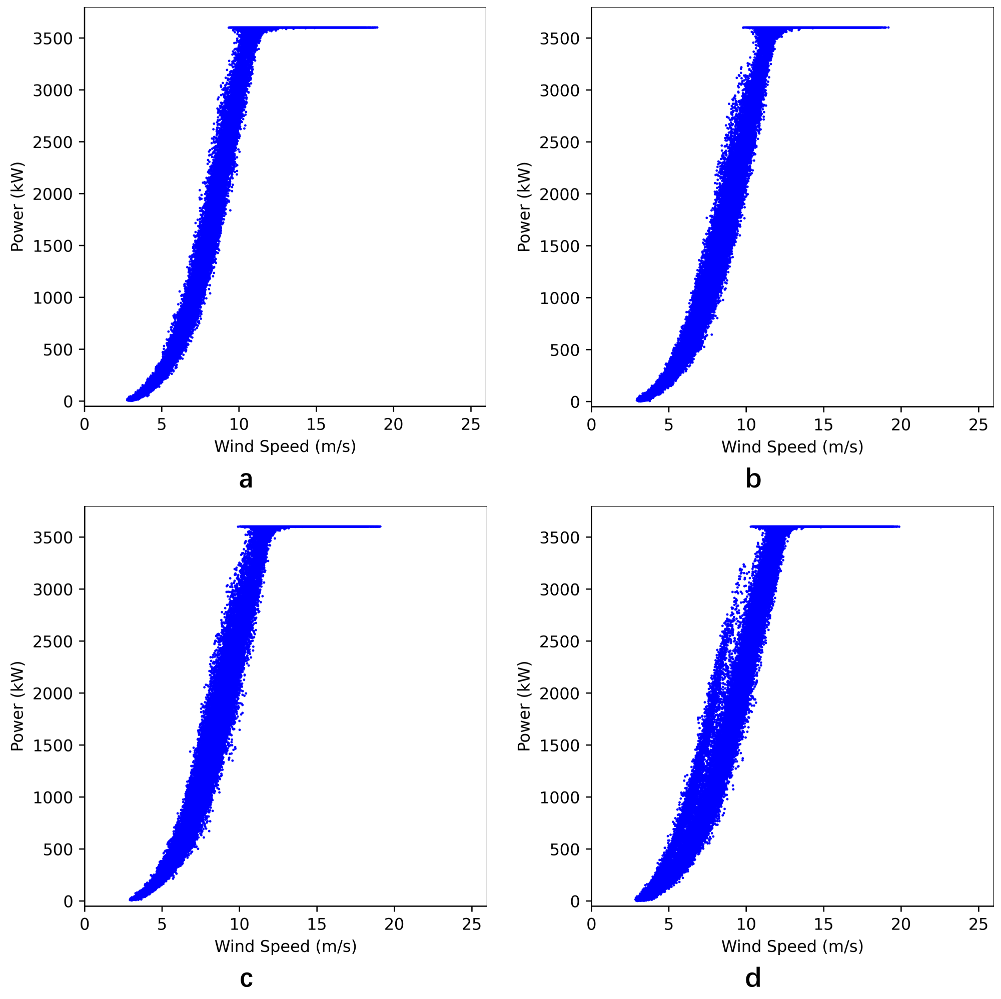


Fig. 8. Power curves processed by iForest: (a), (b), (c), and (d) for WT-1, 2, 3, and 4, respectively.

4.2 Simulation platform

The proposed methodology was implemented using Python with the Keras library packaged in TensorFlow [32]. Adaptive Moment Estimation (ADAM) is selected as the optimization algorithm to train our NSFM RBFNN. The default Keras ADAM configuration was used, with values: learning rate of 0.001, the 1st/2ndmoment decay rates of 0.9/0.999, and epsilon of 10^{-7}. The training and validation losses both used MSE as it was found to have stable compatibility on our datasets. The unknown variables were synaptic weights and one offset, which were iteratively calculated during training. RBF centres with corresponding widths were also iteratively calculated to achieve better approximation after kernel initialization. The hyperparameters of batch size and maximum epochs were 1,000 and 300 respectively.

4.3 Comparison of different methods

4.3.1 Evaluation of modelling accuracy

To evaluate the accuracy of our proposed WTPC modelling, multiple statistical criteria were applied. The Normalized Root Mean Square Error (NRMSE) [33-36], Median Absolute Error (MAE) [14, 37, 38], and coefficient of determination (R^2) [3, 14, 33, 37] were applied and are defined in Eqs. (17), (18), and (19), respectively.

$$MSE = \frac{\sum_{n=1}^{N}(\hat{y}_n - y_n)^2}{N} \tag{15}$$

$$RMSE = \sqrt{MSE} \tag{16}$$

$$NRMSE = {}^{RMSE}/_{\bar{y}} \tag{17}$$

$$MAE = median(|\hat{y}_n - y_n|) \tag{18}$$

$$R^2 = 1 - \frac{\sum_{n=1}^{N}(\hat{y}_n - y_n)^2}{\sum_{n=1}^{N}(y_n - \bar{y})^2} \tag{19}$$

where y_n is the n-th measured output, $\hat{y}_n$ is the n-th model output, and $\bar{y}$ is the mean of measured outputs.

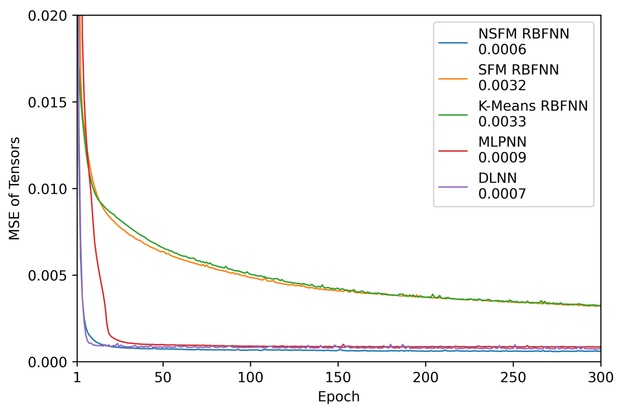
NRMSE is widely utilized as a dimensionless metric, where a large value indicates estimations vary significantly from measurements [33, 37]. MAE is also adopted here to avoid a result influenced by a few instances [14, 38]. R^2 describes the variance of model outputs from measurements [14, 33, 37]. If R^2 is closer to unity, the model will produce a more accurate result, which offers a straightforward way to examine the estimation or prediction accuracy.

4.3.2 Comparison of modelling

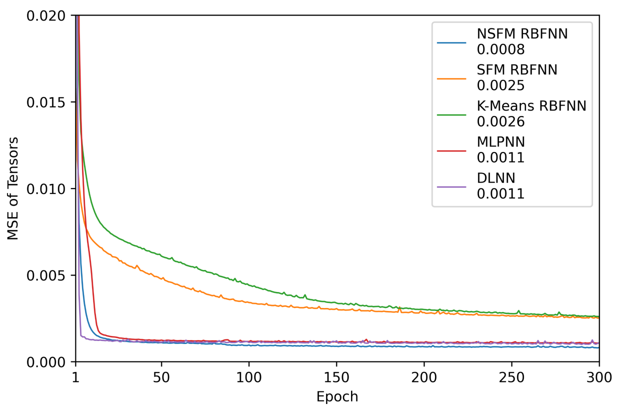
Four additional model types were selected to compare against NSFM RBFNN, these were: SFM RBFNN, K-Means RBFNN, two-layer MLPNN, and DLNN. SFM RBFNN was set to have the same solution space as NSFM

RBFNN. Due to the effect of the number of clusters of K-Means [23], an extensive search for determining the number of clusters was carried out within the range of 50 to 150 with steps of 25. Another extensive search for optimizing MLPNN was also executed, where neurons in the first layer ranged from 30 to 60 with steps of 10, and neurons in the second layer ranged from 10 to 20 with steps of 5. Since deep learning methods are widely applied in operating and controlling WTs [1, 20, 33, 39], this paper also chose the DLNN suggested in [1] for comparison. This DLNN had a 5-layer architecture, in which 20 neurons were in the first and fifth layers and 50 neurons were in the second, third, and fourth layers.

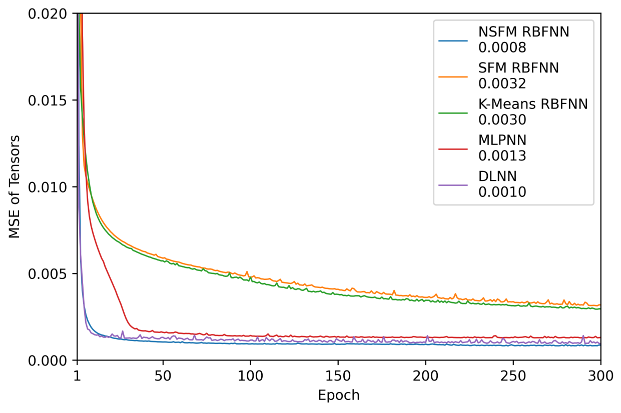
The validation losses in the form of MSE (described in Eq. (15)) of tensors are displayed in **Fig. 9**. NSFM RBFNN and DLNN are shown to converge faster than the others, where both reached steady MSE values before 50 epochs. NSFM RBFNN achieves the minimum validation losses for all datasets. DLNN also had a favourable performance with no significant difference from RBFNN. MLPNN achieved similar validation losses at the end of training but converged at slower speeds. Another advantage of NSFM RBFNN is that the validation loss exhibited smooth training. Besides, with the aid of flexible widths, our proposed NSFM RBFNN has advantages in contrast to SFM and K-Means in terms of training performance.



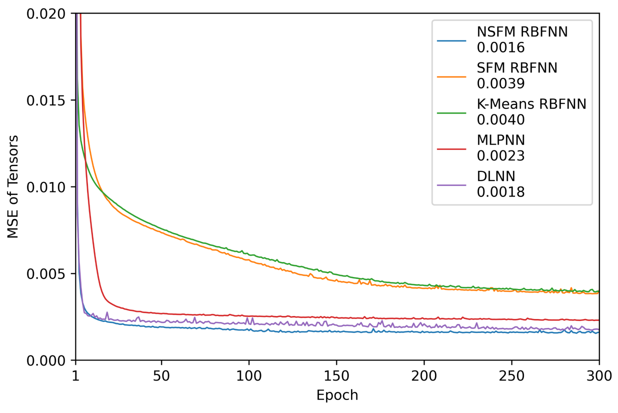
a



b



c



d

Fig. 9. MSE Training loses for the models: (a), (b), (c), and (d) for WT-1, 2, 3, and 4, respectively.

In **Fig. 10,** the degree of fitting of estimations produced by NSFM RBFNN with measurements is shown. The estimations matched the measurements well, where the linear relationship is clear to identify and R^2 averaged 0.9934. Of the WTs, WT-4 faired worst with an R^2 of 0.9890. This is likely a result of WT-4' s WTPC being more scattered than the other turbines after anomaly removal, as shown in **Fig. 8**.

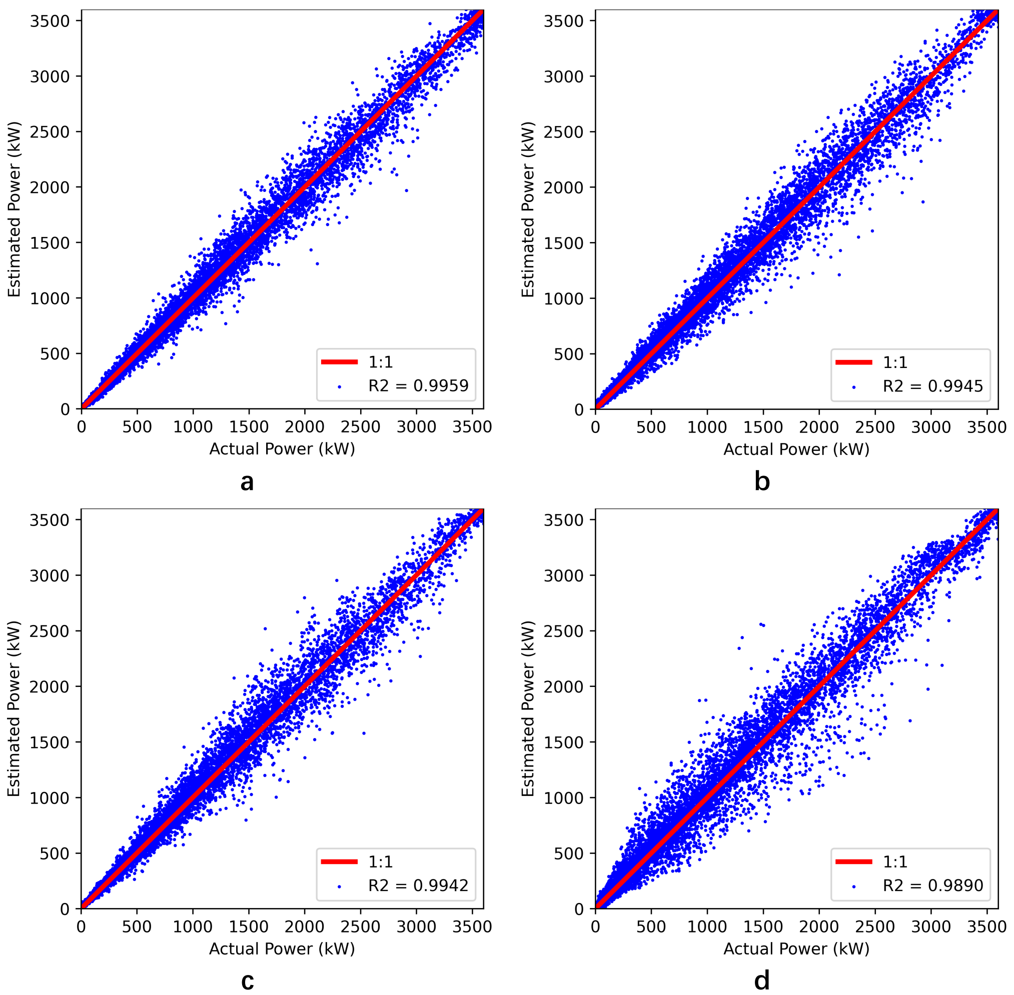


Fig. 10. Estimated versus actual power for NSFM RBFNN: (a), (b), (c), and (d) for WT-1, 2, 3, and 4, respectively.

The statistical results discussed in **Section 4.3.1** for the testing datasets are shown in **Table 4**. In terms of NRMSE and R^2, NSFM RBFNN is the best performer for all datasets, although it is only marginally better compared to MLPNN and DLNN for these metrics. WT-1 achieves the minimum value of NRMSE at 0.0424 and R^2 reaches the maximum value of 0.9959. According to NRMSE and R^2, NSFM RBFNN can provide a more precise group estimation compared to the other methods. Considering MAE, the models of NSFM

RBFNN, MLPNN, and DLNN distinctly outperform SFM RBFNN and K-Means RBFNN, however, none of the three models have advantages over the other two.

According to such metrics, it can be concluded that: i) SFM and K-Means RBFNNs are unable to derive high-accuracy WTPC models, ii) the accuracy of MLPNN degrades faster for scattered WT outputs than NSFM RBFNN and DLNN, iii) the objective of NSFM RBFNN or DLNN is not to reduce the error of each instance but to minimize the error of a group. Due to the similar performance among NSFM RBFNN, MLPNN, and DLNN, a significance test of predictions is carried out in the next section to investigate further.

Table 4

Comparison among the methods regarding NRMSE, MAE, and R^2.

No.	Metrics	NSFM RBFNN	SFM RBFNN	K-Means RBFNN	MLPNN	DLNN
1	Fuzzy partition	[9,10,7,7]	[7,7,7,7]	-	-	-
	No. of nodes	93	53	75	[40,15]	190
	NRMSE	0.0424	0.0958	0.0968	0.0494	0.0478
	MAE	21.7439	129.9651	134.7788	20.2346	18.2813
	R^2	0.9959	0.9790	0.9785	0.9944	0.9948
2	Fuzzy partition	[7,6,7,7]	[7,7,7,7]	-	-	-
	No. of nodes	50	50	50	[60,20]	190
	NRMSE	0.0503	0.0897	0.0912	0.0586	0.0582
	MAE	25.2810	110.8882	115.6800	25.0110	27.0013
	R^2	0.9945	0.9825	0.9819	0.9925	0.9926
3	Fuzzy partition	[8,6,8,7]	[7,7,7,7]	-	-	-
	No. of nodes	61	59	50	[30,20]	190
	NRMSE	0.0510	0.0986	0.0950	0.0637	0.0532
	MAE	25.9001	128.3840	121.1400	26.7006	20.5363
	R^2	0.9942	0.9782	0.9798	0.9909	0.9936
4	Fuzzy partition	[10,6,8,8]	[7,7,7,7]	-	-	-
	No. of nodes	95	59	50	[40,15]	190
	NRMSE	0.0726	0.1145	0.1139	0.0853	0.0747
	MAE	36.0088	130.1140	138.6030	44.9436	37.3907
	R^2	0.9890	0.9726	0.9729	0.9848	0.9883
Mean	NRMSE	0.0541	0.0997	0.0992	0.0643	0.0585
	MAE	27.2334	124.8378	127.5505	29.2225	25.8024
	R^2	0.9934	0.9781	0.9783	0.9907	0.9923

4.4 Diebold-Mariano test

Diebold and Mariano introduced explicit tests of the null hypothesis regarding the difference in accuracy between two forecasting models [40, 41]. There is no quadratic or symmetric requirement for the loss function, and model prediction errors can be non-Gaussian, nonzero-mean, serially correlated, and contemporaneously correlated [40]. As presented in **Section 4.3.2**, the strength of NSFM RBFNN is not overwhelming to MLPNN and DLNN. Therefore, the Diebold-Mariano test will be carried out to investigate further.

Let H_0 be the null hypothesis that there is no difference between NSFM RBFNN and a competitive model, H_1 be the hypothesis that the output power estimated by NSFM RBFNN is better, and H_2 be the hypothesis that the output power estimated by the competitive model is better. The loss function applied in this hypothesis testing is MSE [41]. Let S_0 and p_0 be the statistic and p-value of the Diebold-Mariano test, respectively, and the confidence level is set to 95% [40, 41]. A p_0 value greater than 0.05 indicates there is no difference. If p_0 is less than 0.05 and S_0 is negative, H_1 will be accepted, otherwise, H_2 will be accepted.

The results of the Diebold-Mariano test are shown in **Table 5**. This shows S_0 is always negative and p_0 is always less than 0.05, which demonstrates that the output power estimated by NSFM RBFNN is significantly better than the other compared models.

Table 5

Results of the Diebold-Mariano tests.

No.		SFM RBFNN	K-Means RBFNN	MLPNN	DLNN
1	S_0	-57.7355	-61.0097	-18.1354	-16.0603
	p_0	0.00E+00	0.00E+00	1.30E-72	1.72E-57
	Result	Always accept H_1, decline H_0			
2	S_0	-48.8173	-50.6970	-14.6082	-16.2888
	p_0	0.00E+00	0.00E+00	5.96E-48	4.54E-59
	Result	Always accept H_1, decline H_0			
3	S_0	-52.4504	-51.0306	-19.4932	-6.4624
	p_0	0.00E+00	0.00E+00	1.98E-83	1.07E-10
	Result	Always accept H_1, decline H_0			
4	S_0	-37.0568	-39.3437	-13.0935	-4.0886
	p_0	6.69E-285	0.00E+00	6.57E-39	4.37E-05
	Result	Always accept H_1, decline H_0			

4.5 Robustness

Most evaluations regarding WTPC modelling are carried out on filtered datasets [1, 14, 20, 33]. However, this is not consistent with reality as anomalies are inevitable due to the nature of measuring. As a result, we perform additional tests to investigate the performance of robustness of each method under raw measurements, with no anomaly detection or filtering being performed. Only NSFM RBFNN and DLNN are taken into considerations for the comparison in this section according to the result in **Section 4.4**. To pass numeric data into models, if a parameter of measurement is missing, it will be replaced with the previous value. This is easily implemented and often used in industry as a holder.

Prediction error results for the two methods for each case are presented in the form of box plots in **Fig. 11**. For an ideal WTPC model, the median and mean of prediction error should be close to and symmetrical about 0 kW. According to **Fig. 11**, it is obvious that the medians for both NSFM RBFNN and DLNN are around 0 kW, suggesting both models make reasonable estimates for normal data. However, only the mean of error of NSFM RBFNN is close to 0 kW, whilst for DLNN it averages over 160 kW. This is the result of the compounding of two interrelated factors: poor approximation of the WTPC and type i anomalies. This produced a small number of extremely large error predictions, so producing a large mean prediction error. This serves as an illustration of the ease at which accuracy can be lost when dealing with datasets containing anomalies.

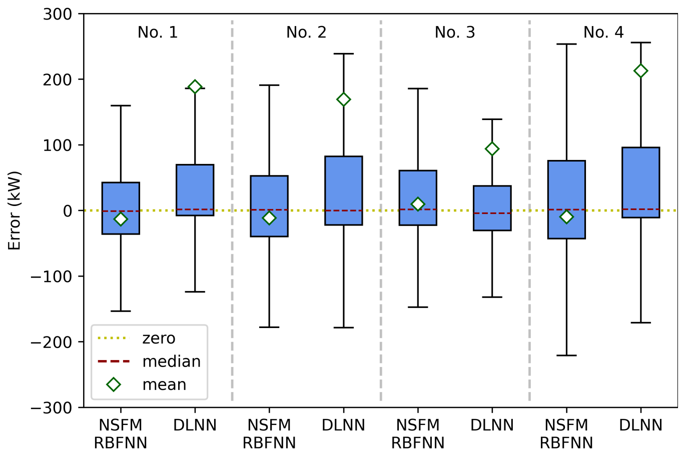


Fig. 11. Error distributions of NSFM RBFNN and DLNN tested on raw datasets.

Table 6

Comparison of NSFM RBFNN and DLNN on raw datasets.

No.	Metrics	NSFM RBFNN	DLNN
1	NRMSE	0.1535	0.3780
	MAE	38.8511	31.8656
	R^2	0.9606	0.7613
	S_0	-73.31	
	p_0	0	
2	NRMSE	0.1514	0.3527
	MAE	46.4843	38.2208
	R^2	0.9610	0.7881
	S_0	-66,78	
	p_0	0	
3	NRMSE	0.1668	0.3006
	MAE	42.3746	33.3178
	R^2	0.9536	0.8492
	S_0	-49.47	
	p_0	0	
4	NRMSE	0.1720	0.4213
	MAE	59.7408	55.6823
	R^2	0.9477	0.6861

	S_0	-83.51	
	p_0	0	
Mean	NRMSE	0.1609	0.3631
	MAE	46.8627	39.7716
	R^2	0.9557	0.7712

As presented in **Table 6**, the accuracy metrics of NRMSE and R^2 are significantly worse for DLNN compared to NSFM RBFNN. NRMSE averages 0.36 for DLNN, approximately 125% greater than that of NSFM RBFNN. Additionally, R^2 drops to 0.77 for DLNN, significantly worse than NSFM RBFNN's 0.96. These results suggest NSFM RBFNN is the more robust of the two models to anomalous inputs. Conversely, the metric of MAE is marginally greater for NSFM RBFNN compared to DLNN, at 46.9 kW and 39.8 kW respectively. However, this is acceptable considering the WTs' installed capacities of 3,600 kW. From the results, including the Diebold-Mariano test, there is no doubt that the output power of offshore WTs forecasted by NSFM RBFNN is significantly more accurate than DLNN for real-world, anomaly-containing data so showing NSFM RBFNN to have good robustness.

5. Conclusion

This paper proposed an NSFM RBFNN based method for offshore WTPC modelling, in which Algorithm 1 provided initial RBF centres and widths and Algorithm 2 optimized the nonsymmetric fuzzy partition for NSFM. Four SCADA datasets were used to compare NSFM RBFNN with other neural network models across multiple metrics and the Diebold-Mariano test. Advantages of our proposed method include fast convergence speed, high accuracy, and robustness. The main conclusions drawn from the case study conducted can be summarized as:

a. SFM RBFNN and K-Means RBFNN are not suitable for offshore WTPC modelling due to their low accuracies, which implies that a radius-based Gaussian kernel is insufficient to capture the characteristics of offshore WT SCADA data.
b. MLPNN cannot be used for estimating WTPC for scattered WT datasets, such as WT-4. DLNN fairs better than MLPNN thanks to its sophisticated architecture, however, performance is still significantly worse compared to that of a cleaned dataset.
c. NSFM RBFNN can deliver accurate and reliable offshore WTPC models. Additionally, this model is more robust to anomalies than DLNN. As proven by the multiple metrics and error distributions, NSFM RBFNN intends to capture characteristics of a group rather than individual instances, which accounts for its robustness.

As a result, our NSFM RBFNN can aid the Energy industry in monitoring and controlling offshore wind turbines by establishing an accurate and robust WTPC model.

Acknowledgement

The authors would like to thank Ørsted for providing the SCADA data.

References

[1] Z. Lin, X. Liu, and M. Collu, "Wind power prediction based on high-frequency SCADA data along with isolation forest and deep learning neural networks," *International Journal of Electrical Power & Energy Systems,* vol. 118, 2020.

[2] I. Pineda, "Wind energy and economic recovery in Europe: How wind energy will put communities at the heart of the green recovery," WindEurope, Oct 2020, [Online]. Available: https://windeurope.org/data-and-analysis/product/wind-energy-and-economic-recovery-in-europe/.

[3] V. Sohoni, S. C. Gupta, and R. K. Nema, "A Critical Review on Wind Turbine Power Curve Modelling Techniques and Their Applications in Wind Based Energy Systems," *Journal of Energy,* vol. 2016, pp. 1-18, 2016.

[4] M. Schlechtingen, I. F. Santos, and S. Achiche, "Using Data-Mining Approaches for Wind Turbine Power Curve Monitoring: A Comparative Study," *IEEE Transactions on Sustainable Energy,* vol. 4, no. 3, pp. 671-679, 2013.

[5] M. Lydia, A. I. Selvakumar, S. S. Kumar, and G. E. P. Kumar, "Advanced Algorithms for Wind Turbine Power Curve Modeling," *IEEE Transactions on Sustainable Energy,* vol. 4, no. 3, pp. 827-835, 2013.

[6] M. Morshedizadeh, M. Kordestani, R. Carriveau, D. S. K. Ting, and M. Saif, "Improved power curve monitoring of wind turbines," *Wind Engineering,* vol. 41, no. 4, pp. 260-271, 2017.

[7] S. Li, D. C. Wunsch, E. A. O’ Hair, and M. G. Giesselmann, "Using neural networks to estimate wind turbine power generation," *IEEE Transactions on Energy Conversion,* vol. 16, no. 3, pp. 276-282, Sep 2001.

[8] M. Carolin Mabel and E. Fernandez, "Analysis of wind power generation and prediction using ANN: A case study," *Renewable Energy,* vol. 33, no. 5, pp. 986-992, 2008.

[9] F. Pelletier, C. Masson, and A. Tahan, "Wind turbine power curve modelling using artificial neural network," *Renewable Energy,* vol. 89, pp. 207-214, 2016.

[10] M. N. Jyothi and P. V. R. Rao, "Very-short term wind power forecasting through Adaptive Wavelet Neural Network," presented at the 2016 Biennial International Conference on Power and Energy Systems: Towards Sustainable Energy (PESTSE), 2016.

[11] R. P. Shetty, A. Sathyabhama, P. P. Srinivasa, and A. A. Rai, "Optimized Radial Basis Function Neural Network model for wind power prediction," presented at the 2016 Second International Conference on Cognitive Computing and Information Processing (CCIP), Mysore, 2016.

[12] P. Zhao, J. Wang, J. Xia, Y. Dai, Y. Sheng, and J. Yue, "Performance evaluation and accuracy enhancement of a day-ahead wind power forecasting system in China," *Renewable Energy,* vol. 43, pp. 234-241, 2012.

[13] J. Liu, X. Wang, and Y. Lu, "A novel hybrid methodology for short-term wind power forecasting based on adaptive neuro-fuzzy inference system," *Renewable Energy,* vol. 103, pp. 620-629, 2017.

[14] D. Karamichailidou, V. Kaloutsa, and A. Alexandridis, "Wind turbine power curve modeling using radial

basis function neural networks and tabu search," *Renewable Energy,* vol. 163, pp. 2137-2152, 2021.

[15] S. AG. (2011). *Siemens Wind Turbine SWT-3.6-120* [Online]. Available: www.siemens.com/energy.

[16] Y. Zhao, J. Pei, and H. Chen, "Multi-layer radial basis function neural network based on multi-scale kernel learning," *Applied Soft Computing,* vol. 82, 2019.

[17] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection," *ACM Computing Surveys,* vol. 41, no. 3, pp. 1-58, 2009.

[18] Y. Zhu, C. Zhu, C. Song, Y. Li, X. Chen, and B. Yong, "Improvement of reliability and wind power generation based on wind turbine real-time condition assessment," *International Journal of Electrical Power & Energy Systems,* vol. 113, pp. 344-354, 2019.

[19] Y. Jiang, L. Guo, and S. You, "Research on nodal wind power values and optimal accommodation based on locational marginal price," *International Journal of Electrical Power & Energy Systems,* vol. 109, pp. 343-350, 2019.

[20] A. Kisvari, Z. Lin, and X. Liu, "Wind power forecasting – A data-driven method along with gated recurrent neural network," *Renewable Energy,* vol. 163, pp. 1895-1909, 2021.

[21] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," presented at the 2008 Eighth IEEE International Conference on Data Mining, 2008.

[22] S. Haykin, *Neural Networks and Learning Machines*, 3rd ed. Pearson Education, Inc., 2009.

[23] S.-K. Oh, W.-D. Kim, W. Pedrycz, and S.-C. Joo, "Design of K-means clustering-based polynomial radial basis function neural networks (pRBF NNs) realized with the aid of particle swarm optimization and differential evolution," *Neurocomputing,* vol. 78, no. 1, pp. 121-132, 2012.

[24] H. Sarimveis, A. Alexandridis, G. Tsekouras, and G. Bafas, "A Fast and Efficient Algorithm for Training Radial Basis Function Neural Networks Based on a Fuzzy Partition of the Input Space," *Industrial & Engineering Chemistry Research,* vol. 41, 4, pp. 751-759, 2002.

[25] A. Alexandridis, E. Chondrodima, and H. Sarimveis, "Radial basis function network training using a nonsymmetric partition of the input space and particle swarm optimization," *IEEE Trans Neural Netw Learn Syst,* vol. 24, no. 2, pp. 219-30, Feb 2013.

[26] A. Alexandridis, H. Sarimveis, and K. Ninos, "A Radial Basis Function network training algorithm using a non-symmetric partition of the input space – Application to a Model Predictive Control configuration," *Advances in Engineering Software,* vol. 42, no. 10, pp. 830-837, 2011.

[27] A. Alexandridis and E. Chondrodima, "A medical diagnostic tool based on radial basis function classifiers and evolutionary simulated annealing," *J Biomed Inform,* vol. 49, pp. 61-72, Jun 2014.

[28] J. Nie, "Fuzzy control of multivariable nonlinear servomechanisms with explicit decoupling scheme," *IEEE Transactions on Fuzzy Systems,* vol. 5, no. 2, pp. 304-311, May 1997 1997.

[29] A. Alexandridis, P. Patrinos, H. Sarimveis, and G. Tsekouras, "A two-stage evolutionary algorithm for variable selection in the development of RBF neural network models," *Chemometrics and Intelligent Laboratory Systems,* vol. 75, no. 2, pp. 149-162, 2005.

[30] A. Hertz and D. d. Werra, "The tabu search metaheuristic: How we used it," *Annals of Mathematics and Artificial Intelligence,* vol. 1, pp. 111–121, September 1990.

[31] M. Kumar, A. Sahu, and P. Mitra, "A comparison of different metaheuristics for the quadratic assignment problem in accelerated systems," *Applied Soft Computing,* vol. 100, 2021.

[32] M. Abadi *et al.*, *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI ' 16)* (TensorFlow: A System for Large-Scale Machine Learning). Savannah, GA: USENIX Association, 2016.

[33] Y.-Y. Hong and C. L. P. P. Rioflorido, "A hybrid deep learning-based neural network for 24-h ahead wind power forecasting," *Applied Energy,* vol. 250, pp. 530-539, 2019.

[34] N. Chen, Z. Qian, I. T. Nabney, and X. Meng, "Wind Power Forecasts Using Gaussian Processes and Numerical Weather Prediction," *IEEE Transactions on Power Systems,* vol. 29, no. 2, pp. 656-665, 2014.

[35] M. G. Lobo and I. Sanchez, "Regional Wind Power Forecasting Based on Smoothing Techniques, With Application to the Spanish Peninsular System," *IEEE Transactions on Power Systems,* vol. 27, no. 4, pp. 1990-1997, 2012.

[36] J. Wang, W. Yang, P. Du, and Y. Li, "Research and application of a hybrid forecasting framework based on multi-objective optimization for electrical power system," *Energy,* vol. 148, pp. 59-78, 2018.

[37] S. Hanifi, X. Liu, Z. Lin, and S. Lotfian, "A Critical Review of Wind Power Forecasting Methods—Past, Present and Future," *Energies,* vol. 13, no. 15, 2020.

[38] C. M. St. Martin, J. K. Lundquist, A. Clifton, G. S. Poulos, and S. J. Schreck, "Wind turbine power production and annual energy production depend on atmospheric stability and turbulence," *Wind Energy Science,* vol. 1, no. 2, pp. 221-236, 2016.

[39] Z. Lin and X. Liu, "Wind power forecasting of an offshore wind turbine based on high-frequency SCADA data and deep learning neural network," *Energy,* vol. 201, 2020.

[40] F. X. Diebold and R. S. Mariano, "Comparing Predictive Accuracy," *Journal of Business & Economic Statistic,* vol. 13, no.3, pp. 253-263, July 1995.

[41] D. Harvey, S. Leybourne, and P. Newbold, "Testing the equality of prediction mean squared errors," *International Journal of Forecasting,* vol. 13, no. 2, pp. 281-291, June 1997.