Fault Detection of Wind Turbine Gearboxes Based on IBOA-ERF

Mingzhu Tang 1,†, Chenhuan Cao 1, Huawei Wu 2,*, Hongqiu Zhu 3,†, Jun Tang 1, Zhonghui Peng 1 and Yifan Wang 1

1 School of Energy and Power Engineering, Changsha University of Science & Technology, Changsha 410114, China
2 Hubei Key Laboratory of Power System Design and Test for Electrical Vehicle, Hubei University of Arts and Science, Xiangyang 441053, China
3 School of Automation, Central South University, Changsha 410083, China
* Correspondence: whw_xy@hbuas.edu.cn
† These authors contributed equally to this work.

Abstract: As one of the key components of wind turbines, gearboxes are under complex alternating loads for a long time, and the safety and reliability of the whole machine are often affected by the failure of internal gears and bearings. Aiming at the difficulty of optimizing the parameters of wind turbine gearbox fault detection models based on extreme random forest, a fault detection model with extreme random forest optimized by the improved butterfly optimization algorithm (IBOA-ERF) is proposed. The algebraic sum of the false alarm rate and the missing alarm rate of the fault detection model is constructed as the fitness function, and the initial position and position update strategy of the individual are improved. A chaotic mapping strategy is introduced to replace the original population initialization method to enhance the randomness of the initial population distribution. An adaptive inertia weight factor is proposed, combined with the landmark operator of the pigeon swarm optimization algorithm to update the population position iteration equation to speed up the convergence speed and improve the diversity and robustness of the butterfly optimization algorithm. The dynamic switching method of local and global search stages is adopted to achieve dynamic balance between global exploration and local search, and to avoid falling into local optima. The ERF fault detection model is trained, and the improved butterfly optimization algorithm is used to obtain optimal parameters to achieve fast response of the proposed model with good robustness and generalization under high-dimensional data. The experimental results show that, compared with other optimization algorithms, the proposed fault detection method of wind turbine gearboxes has a lower false alarm rate and missing alarm rate.

Keywords: fault detection; butterfly optimization algorithm; extreme random forest; wind turbine; gearbox

1. Introduction

As an important source of clean and renewable energy, wind energy resources play an important role in the sustainable development of the national economy. The use of wind power is very environmentally friendly, and wind energy reserves are huge, so wind power is attracting more and more attention from countries all over the world. According to the forecast of the Global Wind Energy Council (GWEC), global wind power will increase by 557 GW in the next five years (2022–2026), with a compound annual growth rate of 6.6%. By 2026, the globally newly installed capacity of wind power will reach 128.8 GW, of which the newly installed capacity of onshore wind power will be 97.4 GW, while the newly installed capacity of offshore wind power will be 31.4 GW [1]. However, abundant wind resources are often found in remote areas, and the occurrence of some extreme weather conditions can lead to the failure of wind turbines [2]. Compared with the tower base, the
narrow nacelle does not have a solid foundation, and the factors of power matching and torsional deformation in the drive train are always concentrated in a weak link. Much research has proven that this link is often the gearbox in the unit [3]. The gearbox is an essential mechanical component, and its main purpose is to transport the power generated by the blades to the power generator in order to obtain the appropriate speed [4]. Due to its special installation position, once a fault occurs, it is very difficult to repair. Compared with other unit components, the gearbox has the longest downtime and repair time due to failure, resulting in long-term gearbox downtime. Therefore, providing accurate guidance at the first instance of failure can reduce the operating cost and maintenance cost of the wind turbine, which has great economic and engineering value [5]. In recent years, scholars have carried out extensive applied research on the fault detection of wind turbines.

Currently, research on fault detection of wind turbine gearboxes mainly includes methods based on signal processing, along with data-driven and model-based methods [6–8]. Signal-based approaches—such as spectral analysis, wavelet transform [9], and non-parametric spectrum estimation—are often carried out. However, for stationary signal power, unlike the theoretical infinite-length signal, the actual observed signal is a finite-length signal. Low resolution of frequency is inevitable in the conversion process. The data-driven approach requires large volumes of historical data and multidimensional features [10]. Today, machine-learning-based fault detection approaches are used extensively in the field of industry [11].

In machine learning, the decision tree classification model is a tree structure, which is strongly intuitive and easy to understand, and has become a popular technology of online detection. Liang [12] proposed to encrypt the decision table using a searchable symmetric encryption method to improve the classification speed and solve the detection requirement in microseconds. Stetco [13] reviewed the machine learning methods used in wind turbine blades, generator temperature fault detection, etc. Classification is mostly used when using SCADA datasets or simulation data, and decision trees are the most commonly used models. In general, decision trees are prone to overfitting and poor generalization performance, and small changes in the data may lead to the generation of completely different trees—that is, their stability performance needs to be improved. To solve this problem, Feng [14] used the adaptive boost algorithm to find the mapping between incoming data and outgoing data, and the overall accuracy of the model was improved.

The boost algorithm in machine learning refers to integrating multiple weak classifiers to reduce the time complexity of a single decision tree and make the model easy to display [15]. Liu [16] proposed a fault detection method based on NFSW-BP-AdaBoost to evaluate the combination of multiple classifiers with non-fuzzy solution coefficients to improve the recognition rate of faults. Chakraborty [17] designed the data-driven model of extreme gradient boosting (XGBoost), using the dynamic adjusted threshold to judge the occurrence of faults, which improved the quality of the model and had strong generalization ability. Xu [18] designed cost-sensitive GBDT (CS-GBDT) to improve the problem of low diagnostic accuracy in the face of unbalanced datasets, and used multiple-domain feature extraction and feature selection to enhance diagnostic accuracy. However, in the face of high-dimensional complex data in actual wind farms, the boost algorithm consumes too much memory, making it easy to reduce the calculation accuracy and fault detection accuracy.

Owing to the large amount of data and high dimensionality of real wind farms, existing studies usually have problems such as poor performance and long training time. Extreme random forest is an ensemble tree algorithm with complete randomness proposed on the basis of decision trees. The feature values are selected for segmentation in the training phase to obtain the segmentation values. This method has strong randomness, and in practical applications it shows high accuracy in high-dimensional datasets, can easily achieve parallelization, and has strong generalization performance. However, in the domain of practical fault detection, the selection of hyperparameters is extremely critical to
the final detection results, and suitable hyperparameters can prevent the local convergence of the model and achieve the best results [19].

For high-dimensional nonlinear problems, the modern intelligent optimization algorithm is widely used in the field of fault detection [20]. In practical applications, the optimization algorithm is used to find the optimal scheme or parameter value among many schemes or parameter values, so that some performance and function indices of the system can reach optimal values. Arora [21] introduced a new nature-inspired heuristic algorithm—the butterfly optimization algorithm, which has the strengths of requiring few adjustment parameters and strong convergence. However, in the face of complex optimization problems such as high-dimensional data, it is prone to being trapped in local optima, and another problem is its slow convergence speed [22].

In view of the above problems, a fault detection model with extreme random forest optimized by improved butterfly optimization algorithm (IBOA-ERF) was proposed. In the improved butterfly optimization algorithm, chaotic mapping is introduced to initialize the population, and the adaptive inertia weight factor is introduced. Combined with the pigeon swarm optimization algorithm, adaptive dynamic switching is proposed to control the conversion of the search stage, which is integrated into the population position update formula, and the convergence speed and optimization accuracy are greatly improved. Firstly, the data are cleaned using Pearson’s correlation analysis, reducing the data’s dimensions and deleting redundant features. Secondly, the sample dataset is divided into two categories: a training set and a test set. The improved butterfly algorithm is used to generate the best hyperparameters of the extreme random forest, and the IBOA-ERF fault detection model is constructed to detect the gearbox faults of wind turbines.

2. Fault Detection of Wind Turbine Gearboxes

As one of the most significant structural parts of a wind turbine, the gearbox is subject to very complex forces, and works under complex alternating loads and harsh working environments for a long time.

Figure 1 shows schematic diagrams of a wind turbine’s structure and the fault detection process. When the unsteady wind acts on the unit, different loads are generated [23]. The blade produces axial thrust and circumferential shear, resulting in deflection movement [24]. The torsional main bearing transmits the blade torque to the gearbox to complete the output of the corresponding load. In the generator, the torque on the motor shaft continuously cuts the magnetic induction line to output power, and completes the conversion of wind energy, mechanical energy, and power [25]. Subsequently, the coordination of major electrical parameters and data interaction is completed through the frequency converter and control unit. The actual operating data of the wind turbine are stored in the SCADA system, making it easy to extract data for fault detection.

The proportion of failures caused by broken teeth, pitting, gluing, and wear of gears inside the gearbox is about 60%, while the proportion of failures caused by damage to bearings such as burns, balls falling off, and cage deformation is about 20%, which seriously impact the security and stability of the whole machine’s operation [26]. Due to the high fault dimensions and redundant parameters, it is important to mine the fault characteristics of gearboxes deeply and determine the fault location and category quickly and accurately for the secure and stable operation of wind turbines.

In summary, in order to further enhance the stability and precision of wind turbine gearbox fault detection, aiming at the problems of gearbox fault data dimension reduction, feature selection, and model parameter optimization, combined with extreme random forest with excellent classification performance, a wind turbine fault detection model based on IBOA-ERF is adopted, which improves the detection precision of the model and ensures the safe operation of the wind turbine.
3. Extreme Random Forest

Random forest (RF) consists of a series of decision trees. The decision tree is a tree structure, in which each internal node represents a categorical judgment, and each leaf node at the bottom represents a classification result; this is detailed in Figures 2 and 3. A subset of $n$ samples of the same size as the sample set is obtained by randomly selecting the sample set. Next, several weak classifiers are built. A decision tree is a tree classification method derived from the training samples by using a set of random vectors.

![Figure 1. Schematic diagrams of a wind turbine’s structure and the fault detection process.](image1)

![Figure 2. Structure diagram of ERF.](image2)
At the time of node-splitting, through top-down recursion, traversing each feature and each value of each feature, and use evaluation criteria such as the Gini coefficient to determine the optimal features and feature values as node features and thresholds. The process iteratively splits down until the entropy of each leaf node is reduced to 0—that is, the class confusion degree of the sample is 0—and then votes to determine the final classification. Through the above steps, the unique path of each sample is determined, and the category of the sample is the category corresponding to the leaf node of the unique path.

While inheriting the good performance of RF, extreme random forest (ERF) has two main differences: First, the original dataset is used in the training set of each decision tree. Due to the randomness of feature selection and node splitting, the obtained results will be better than those of RF. Second, after picking the segmentation features, RF selects an optimal feature value for segmentation, while the ERF splits the randomly selected eigenvalues, which enhances the generic performance of the model, while the size of the decision tree increases. Figure 2 shows a structural diagram of ERF.

The class attribute is determined by the vote of all decision trees, and its vote is based on Equation (1). The larger the calculated $P$, the higher the probability of belonging to the corresponding category. Equation (2) is the voting mechanism principle of the final decision tree. The above method is used to generate the extreme random forest decision tree.

$$P(c|f_i) = \frac{1}{D} \sum_{i=1}^{D} P_t(c|V_i)$$  \hspace{1cm} (1)

$$\hat{c} = \arg\max_c P(c|V_i)$$  \hspace{1cm} (2)

where $V_i$ denotes the feature vector of the sample, $c$ is some kind of category, $D$ denotes the number of trees in the ERF, $P_t(c|V_i)$ denotes the probability that the sample belongs to category $c$ conditional on the feature vector $V_i$, $P(c|V_i)$ is the average value in the ERF, and $\hat{c}$ represents the category corresponding to the maximum value of $P(c|V_i)$.

During the node-splitting phase, for the process of selecting the obtained feature as the splitting feature, Equation (3) is used to measure the score. When the leaf nodes are split, the splitting feature is selected as the feature with the highest score. Samples smaller than the splitting threshold are put in the left leaf node after splitting; otherwise, they are placed in the right leaf node. These procedures are repeated until the sample confusion in the leaf node is 0. Figure 3 illustrates the splitting architecture of the ERF fault tree.

$$\text{Score}_k = \frac{2l_k}{H_k + H_c}$$  \hspace{1cm} (3)
where Score\(_k\) represents the score measurement of the calculated feature, and I\(_k\) denotes the mutual information of the two subsets of the node after splitting on the basis of the corresponding features and splitting threshold of the sample category. H\(_k\) denotes the split entropy of feature k, while H\(_c\) represents the information entropy of the node for the corresponding category.

The choice of hyperparameters in ERF has a great influence on the classification precision of the model, and the optimization of the parameters is difficult. Therefore, optimization algorithms must be introduced to search for the best parameters to enhance the reliability of the fault detection model.

4. Butterfly Optimization Algorithm

In nature, butterflies use their high sensitivity to fragrance to search for food and partners. In 2019, Arora [21] proposed the butterfly optimization algorithm (BOA), which imitates the movements of butterflies in search of food and mating.

4.1. Basic Theory of the Butterfly Optimization Algorithm

Studies have shown that butterflies can accurately determine the location of food by detecting different flavors and flavor intensity during predation [27]. In the butterfly optimization algorithm, each butterfly produces a certain intensity of fragrance according to its fitness, and when it perceives that the fragrance emitted by another butterfly in a certain region is stronger, it will try to approach this butterfly, which is known as global search. When a butterfly perceives its own fragrance to be more intense than that of other butterflies, it will be able to freely move in space, which is known as local search [28].

In the BOA, butterfly fragrance calculation is as shown in Equation (4):

\[
f = sI^{\alpha}
\]

where \(f\) is the fragrance intensity, \(I\) is the stimulus intensity, \(s\) is the sensory modality with a value of 0.01, and \(\alpha\) is the power exponent with a value of 0.1.

In the BOA, the stimulus intensity \(I\) of the individual is influenced by the objective function, and the power exponent \(\alpha\) is the exponent of the increase in fragrance intensity. The transitions of the global and local search stages are controlled by the switching transition frequency \(p \in [0, 1]\). In the global search phase, the position is updated as shown in Equation (5):

\[
x_{i}^{t+1} = x_{i}^{t} + (r_{2} \times g^{*} - x_{i}^{t}) \times f_{i}
\]

where \(x_{i}^{t+1}\) and \(x_{i}^{t}\) are the location information of the i-th individual in the t+1-th and t-th iterations, respectively; \(g^{*}\) is the best value in the current iteration; \(f_{i}\) is the fragrance intensity emitted by the i-th individual; and \(r\) is the random value from 0 to 1. In the local search phase, the position is updated as shown in Equation (6):

\[
x_{i}^{t+1} = x_{i}^{t} + (r_{2} \times x_{j}^{t} - x_{k}^{t}) \times f_{i}
\]

where \(j\) and \(k\) are the random numbers generated in each iteration, while \(x_{j}^{t}\) and \(x_{k}^{t}\) are the location information of the j-th and k-th individuals in the current iteration, respectively.

4.2. Improvement and Innovation of the Butterfly Optimization Algorithm

Compared with some existing meta-heuristic algorithms, the BOA is relatively novel, with simple operation, few parameters to be adjusted, and better robustness. It is superior to some classic intelligent optimization algorithms in terms of optimization ability, and has achieved good results in the preliminary application of engineering practice. However, in the face of complex conditions, its performance is not good, and there are still problems such as its tendency to become trapped in local optima and its low convergence precision.
when solving high-dimensional functions. To solve this problem, the improved butterfly optimization algorithm (IBOA) is constructed through the following four modifications:

1. Introduce a chaotic map to randomly initialize the population position, so that the initial population is random and aperiodic, so as to prevent the exploration process from ending up in a local optimum.
2. Design an adaptive inertia weight factor and apply it to the position update formula to enhance the capability of local search and accelerate the search rate.
3. Introduce the landmark operator sub-item of the pigeon group optimization algorithm, design a new position update formula, enhance the global search capability, and improve the diversity and robustness of the butterfly optimization algorithm.
4. Design a new dynamic switching method for the local search phase and the global search phase, and introduce the variant of trigonometric function as the switching basis, which can effectively prevent trapping in local optima and accelerate the convergence speed.

4.2.1. Chaos Map Initialization

BOA randomly initializes the population position, but using this approach to generate the initial population may lead to uneven distribution and superposition of individual butterfly positions. In the butterfly population, the small change in the initial distribution has a great impact on the subsequent iterative search process. To solve this problem, chaotic variables are used to optimize the search so as to evenly distribute the initial population [29], which can improve the diversity of BOA, greatly improve the convergence speed and optimization accuracy, and prevent premature convergence. After testing and comparison, the classical logistical chaotic mapping is used to initialize the population. The logistic map described in [30] is used to map the variables into the chaotic variable space, and then used the linear transformation to map the generated chaotic variables into the solution space in need of optimization. Figure 4 shows the comparison between the initialization using chaotic mapping and the original initialization method. The specific expression of the logistic map is as shown in Equation (7):

\[
X(t + 1) = X(t) \times \mu \times (1 - X(t)) \quad \mu \in [0, 4], \; X \in [0, 1]
\]  

(7)

where \( \mu \) is the logistics parameter, \( X \) is the position parameter, and \( t \) is the value of the iterations. The research shows that when \( \mu \) is 4, the range of \( X \) is almost evenly distributed in the entire region of 0 to 1, so the value of \( \mu \) in this case is 4.

**Figure 4.** (a) The distribution after random initialization; (b) the distribution after initialization of the chaotic map.
4.2.2. Adaptive Inertia Weighting Factor

According to the basic principle of the BOA, each individual updates or randomly moves its position according to the current best individual position. Therefore, the position of individual butterflies is not fully utilized, and it is easy to become trapped in a local optimum. When the inertia factor is large, the global search capability is strong, and vice versa. Therefore, to address this issue, an adaptive inertia weighting factor was designed to apply to the position update formula, so that the historical optimal position information of the individual is fully utilized. Meanwhile, as the iterations grow in size, the direction and distance of the individual are effectively controlled, so as to enhance the optimization precision and convergence velocity, and avoid falling into local optima. The expression of the inertia weighting factor is as follows:

$$\omega(t) = 1 - \sin \left( \frac{\pi t}{\sqrt{e + 1} \times T_{iter}} \right)$$  \hspace{1cm} (8)

where $\omega$ is the adaptive inertia weight, $T_{iter}$ is the largest value of the number of iterations $t$ in the optimization process, and $e$ is the Euler number.

The position update formula for the global search phase after the introduction of the adaptive inertia weighting factor in BOA is as follows:

$$x_{i}^{t+1} = \omega(t) \times x_{i}^{t} + \left( r^2 \times g^* - x_{i}^{t} \right) \times f_{i}$$  \hspace{1cm} (9)

The position update formula for the local search phase is as follows:

$$x_{i}^{t+1} = \omega(t) \times x_{i}^{t} + \left( r^2 \times x_{j}^{t} - x_{k}^{t} \right) \times f_{i}$$  \hspace{1cm} (10)

4.2.3. Pigeon-Inspired Optimization Algorithm Landmark Operator

Inspired by the nesting activity of pigeons, a new population intelligence optimization algorithm—the pigeon-inspired optimization (PIO) algorithm—was first proposed by Duan [31] in 2014.

PIO simulates pigeon homing using different search mechanisms at different stages. The algorithm includes two models: a compass model and a landmark model. In the compass model, the individual updates the location according to its previous location information and the current global optimal location information. In the landmark operator, on the basis of halving the number of groups in each iteration, the pigeons accelerate the convergence rate according to the average value of group fitness. PIO has the characteristics of fast convergence and high search accuracy, and has been widely used in different fields [32].

The landmark model of PIO is as follows:

$$x_{i}^{t+1} = x_{i}^{t} + r \times (x_{c}^{t} - x_{i}^{t})$$  \hspace{1cm} (11)

$$x_{c}^{t} = \frac{\sum (x_{i}^{t} \times \text{Fit}(x_{i}^{t}))}{N_{p}^{t} \times \sum \text{Fit}(x_{i}^{t})}$$  \hspace{1cm} (12)

$$N_{p}^{t+1} = \frac{N_{p}^{t}}{2}$$  \hspace{1cm} (13)

where $x_{c}^{t}$ is the position of the center of the flock in the current iteration, $\text{Fit}(x_{i}^{t})$ is the value of the fitness function of the i-th pigeon, and $N_{p}^{t}$ is the number of individuals. Other variables are defined as in Equation (5).

In the BOA, the fragrance of butterflies plays an important role in guiding individuals to move to the optimal solution. However, if the population falls into the local optimal position, it is prone to resulting in a stagnant search that does not lead to a globally optimal resolution. Based on this problem, inspired by PIO, combined with the landmark model, a
new butterfly position update formula was constructed. Since the landmark model needs to calculate the average fitness of the group, compared with the compass model, not only is the global search capability greatly enhanced, but also the convergence velocity is improved. The improved butterfly position global search stage update formula is as follows:

\[
x_{i}^{t+1} = \omega(t) \times x_{i}^{t} + \left( r \times x_{j}^{t} - x_{k}^{t} \right) \times f_{i} + r \times (x_{i}^{t} - x_{i}^{t})
\]

(14)

4.2.4. Adaptive Dynamic Switching

In the BOA, the switching between the local search stage and the global search stage is controlled by the switching frequency \( p \). The higher the value of the parameter \( p \), the greater the proportion of global search; the lower the value of \( p \), the greater the proportion of the local search. The value of \( p \) plays a key role in the subsequent search efficiency and convergence rate. To solve this problem, an adaptive dynamic switching frequency strategy is proposed. The oscillation trigonometric function is introduced. The proportion of local and global search stages is dynamically adjusted according to the number of iterations. The random selection search phase is changed in such a way that global search is performed in the early stage, while local search is performed in the middle and late stages.

\[
S_{1}(t) = (t + 1) \times \sin(w t)
\]

(15)

\[
S_{2}(t) = \sqrt{e - \varnothing} \times ((T_{\text{iter}} - t) + 1) \times \sin(w \times (T_{\text{iter}} - t))
\]

(16)

where \( w \) and \( \varnothing \) take the values 100\( \pi \) and 2.55, respectively, while \( e \) is the Euler number. The iterative process, as shown in Figure 5, enters the local search phase when \(|S_{1}(t)| > |S_{2}(t)|\), and otherwise enters the global search phase, which can be experimentally proven to converge faster and search more efficiently.

Figure 5. Switching between global and local search phases.

4.3. Simulation Experiments

In order to verify that the IBOA has better performance in terms of convergence and robustness, a performance comparison experiment was carried out based on six test functions: F1–F3 are unimodal functions to test the convergence performance of the algorithm, while F4–F6 are complex multimodal functions to test global optimization and jump out of local optimization performance. The standard test function information is shown in Table 1.

In order to sufficiently validate the effectiveness of the IBOA, the comparative experiments were conducted with moth–flame optimization (MFO) [33], multi-verse optimization...
(MVO) [34], the sine–cosine algorithm (SCA) [35], the salp swarm algorithm (SSA) [36], and the BOA. The number of iterations was 500, and each method was run 30 times separately on each test function to prevent bias in the outcomes due to random factors, as detailed in Table 2.

Table 1. Basic test function information.

| Function Types | Expressions | Scope | Optimal Value |
|----------------|-------------|-------|---------------|
| **Unimodal**   |             |       |               |
| F1             | $F_1(x) = \sum_{i=1}^{n} x_i^2$ | $[-100, 100]$ | 0            |
| F2             | $F_2(x) = \sum_{i=1}^{n} \left( \frac{\sum_{j=1}^{m} y_j^2}{n} \right)$ | $[-100, 100]$ | 0            |
| F3             | $F_3(x) = \max\{|x_i, 1 \leq i \leq n|\}$ | $[-100, 100]$ | 0            |
| **Multimodal** |             |       |               |
| F4             | $F_4(x) = \sum_{i=1}^{n} [x_i^2 - 10 \cos(2\pi x_i) + 10]$ | $[-5, 12, 5, 12]$ | 0            |
| F5             | $F_5(x) = -20 \exp \left( \sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}} - \frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i) + 10 + e \right)$ | $[-32, 32]$ | 0            |
| F6             | $F_6(x) = \frac{1}{10} \sum_{i=1}^{n} x_i^2 - \frac{1}{\prod_{i=1}^{n} x_i} + 1$ | $[-600, 600]$ | 0            |

Table 2. Experimental results of the test functions.

| Functions | Algorithms | Optimal Value | Worst Value | Average Value | Standard Deviation |
|-----------|------------|---------------|-------------|---------------|--------------------|
| **F1**    | IBOA       | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ |
|           | MFO        | $6.94 \times 10^{-1}$ | $1.00 \times 10^{-1}$ | $1.35 \times 10^{-1}$ | $3.45 \times 10^{-1}$ |
|           | MVO        | $5.97 \times 10^{-1}$ | $1.98 \times 10^{-1}$ | $1.26 \times 10^{-1}$ | $3.39 \times 10^{-1}$ |
|           | SCA        | $6.02 \times 10^{-2}$ | $3.30 \times 10^{-2}$ | $2.98 \times 10^{-2}$ | $6.89 \times 10^{-2}$ |
|           | SSA        | $3.05 \times 10^{-8}$ | $3.04 \times 10^{-6}$ | $2.70 \times 10^{-7}$ | $5.77 \times 10^{-7}$ |
|           | BOA        | $1.08 \times 10^{-11}$ | $1.45 \times 10^{-11}$ | $1.28 \times 10^{-11}$ | $8.81 \times 10^{-13}$ |
| **F2**    | IBOA       | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ |
|           | MFO        | $8.24 \times 10^{-5}$ | $6.67 \times 10^{-3}$ | $1.11 \times 10^{-1}$ | $2.29 \times 10^{-3}$ |
|           | MVO        | $2.44 \times 10^{-2}$ | $2.48 \times 10^{-1}$ | $1.07 \times 10^{-1}$ | $5.59 \times 10^{-2}$ |
|           | SCA        | $8.32 \times 10^{-9}$ | $6.78 \times 10^{-2}$ | $7.90 \times 10^{-3}$ | $1.74 \times 10^{-2}$ |
|           | SSA        | $3.09 \times 10^{-9}$ | $1.91 \times 10^{-6}$ | $1.50 \times 10^{-7}$ | $3.78 \times 10^{-7}$ |
|           | BOA        | $9.46 \times 10^{-12}$ | $1.32 \times 10^{-11}$ | $1.12 \times 10^{-11}$ | $9.03 \times 10^{-13}$ |
| **F3**    | IBOA       | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ |
|           | MFO        | $2.93 \times 10^{-3}$ | $1.04 \times 10^{-1}$ | $2.42 \times 10^{-1}$ | $3.35 \times 10^{-1}$ |
|           | MVO        | $3.34 \times 10^{-2}$ | $1.98 \times 10^{-1}$ | $9.98 \times 10^{-2}$ | $4.08 \times 10^{-2}$ |
|           | SCA        | $4.84 \times 10^{-7}$ | $2.99 \times 10^{-2}$ | $2.38 \times 10^{-3}$ | $6.92 \times 10^{-3}$ |
|           | SSA        | $1.47 \times 10^{-5}$ | $1.02 \times 10^{-4}$ | $2.82 \times 10^{-5}$ | $1.94 \times 10^{-5}$ |
|           | BOA        | $4.29 \times 10^{-9}$ | $6.23 \times 10^{-9}$ | $5.35 \times 10^{-9}$ | $4.60 \times 10^{-10}$ |
| **F4**    | IBOA       | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ |
|           | MFO        | $8.95 \times 10^{0}$ | $8.46 \times 10^{1}$ | $2.47 \times 10^{1}$ | $1.61 \times 10^{1}$ |
|           | MVO        | $4.98 \times 10^{0}$ | $3.38 \times 10^{1}$ | $1.54 \times 10^{1}$ | $7.15 \times 10^{0}$ |
|           | SCA        | $0.00 \times 10^{0}$ | $1.27 \times 10^{1}$ | $6.42 \times 10^{-1}$ | $2.58 \times 10^{0}$ |
|           | SSA        | $3.98 \times 10^{0}$ | $4.18 \times 10^{1}$ | $1.80 \times 10^{1}$ | $8.62 \times 10^{0}$ |
|           | BOA        | $5.54 \times 10^{0}$ | $5.61 \times 10^{1}$ | $3.35 \times 10^{1}$ | $1.94 \times 10^{1}$ |
| **F5**    | IBOA       | $8.88 \times 10^{-16}$ | $8.88 \times 10^{-16}$ | $8.88 \times 10^{-16}$ | $8.88 \times 10^{-16}$ |
|           | MFO        | $1.13 \times 10^{0}$ | $2.00 \times 10^{1}$ | $1.11 \times 10^{1}$ | $8.60 \times 10^{0}$ |
|           | MVO        | $1.03 \times 10^{0}$ | $3.36 \times 10^{0}$ | $1.92 \times 10^{0}$ | $4.99 \times 10^{-1}$ |
|           | SCA        | $3.53 \times 10^{-2}$ | $2.03 \times 10^{1}$ | $1.11 \times 10^{1}$ | $9.62 \times 10^{0}$ |
|           | SSA        | $1.92 \times 10^{-1}$ | $4.62 \times 10^{0}$ | $2.65 \times 10^{0}$ | $8.91 \times 10^{-1}$ |
|           | BOA        | $4.49 \times 10^{-9}$ | $6.87 \times 10^{-9}$ | $6.01 \times 10^{-9}$ | $5.29 \times 10^{-10}$ |
| **F6**    | IBOA       | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ | $0.00 \times 10^{-1}$ |
|           | MFO        | $4.68 \times 10^{-2}$ | $3.49 \times 10^{-1}$ | $1.52 \times 10^{-1}$ | $7.42 \times 10^{-2}$ |
|           | MVO        | $1.35 \times 10^{-1}$ | $5.77 \times 10^{-1}$ | $3.32 \times 10^{-1}$ | $1.20 \times 10^{-1}$ |
|           | SCA        | $6.02 \times 10^{-13}$ | $4.49 \times 10^{-1}$ | $8.19 \times 10^{-2}$ | $1.34 \times 10^{-1}$ |
|           | SSA        | $6.64 \times 10^{-2}$ | $6.44 \times 10^{-1}$ | $2.46 \times 10^{-1}$ | $1.49 \times 10^{-1}$ |
|           | BOA        | $4.63 \times 10^{-14}$ | $1.75 \times 10^{-11}$ | $7.23 \times 10^{-13}$ | $3.18 \times 10^{-12}$ |
To visually demonstrate the optimized capabilities of the IBOA, the iterative graph of the convergence curve of the six benchmark functions was selected, as shown in Figure 6.

![Convergence curves](image_url)

*Figure 6. (a) Convergence curve of function F1; (b) convergence curve of function F2; (c) convergence curve of function F3; (d) convergence curve of function F4; (e) convergence curve of function F5; (f) convergence curve of function F6.*
4.4. Analysis of Simulation Experiment Results

When solving the minimum value problem, the average value is used to evaluate the optimal ability and convergence precision, the standard deviation is used to evaluate the robustness, and the best value and the worst value are used to evaluate the quality of the feasible solution of the algorithm.

As shown in Table 2, in terms of optimal values, the IBOA does not significantly improve in the F5 function, but it still has a great progress trend compared with the basic BOA, and the optimal value is found in other functions, indicating that the initialization of the population position through chaotic mapping maintains the diversity of the algorithm.

From an average perspective, the IBOA's performance is far superior to that of other algorithms, especially in the unimodal function, indicating that the new location update equation combined with the pigeon swarm algorithm and the strategy of dynamic search-stage switching not only accelerates the convergence speed, but also further enhances the quality of the refined search at a later stage, and greatly improves the overall optimization ability.

From the perspective of standard deviation, the capability of the IBOA is significantly superior to that of other methods; the optimization ability is significantly enhanced, and the quality of the IBOA's feasible solutions is high, indicating that the introduction of the adaptive inertia weighting factor strategy in the position update equation effectively maintains the population diversity, improves the global optimization ability, and maintains strong robustness throughout the search process, so as to acquire the global optimal solution.

5. ERF Fault Detection Model Based on the IBOA

5.1. Data Pre-Processing

The operation process of the wind turbine gearbox is complex, the state quantity generated is complex, and there are many redundant variables, increasing the complexity of model training and affecting the prediction performance of the model [37]. As illustrated in Figure 7b, it is important that the data gathered from the SCADA dataset undergo preliminary data cleaning, and then Pearson's correlation analysis is performed to remove redundant feature values [38]. Pearson's correlation coefficient is illustrated in Equation (17):

\[ \rho_{X,Y} = \frac{\text{cov}(X,Y)}{\sigma_X \times \sigma_Y} \]  

where \( \rho \) represents the correlation coefficient between features in the sample, \( \sigma \) represents the standard deviation of the corresponding features, and \( \text{cov} \) represents the covariance between features.

Pearson's correlation coefficient is the upgrade of Euclidean distance, and provides standard data input for the wind turbine gearbox fault detection model. Through Pearson's correlation analysis, redundant features with low partial correlation are removed, making the model training more efficient and the prediction results more accurate [39].

5.2. ERF Fault Detection Model Flowchart and Pseudocode Based on the IBOA

After Pearson's correlation analysis, the dataset is divided into two categories: the training dataset is utilized to train the classification model, while the test dataset is utilized for the prediction of the model, measuring the performance and classification ability of the model, and evaluating the model's prediction performance.

The optimization of the IBOA parameters is shown in Figure 7. Firstly, the position and sensory mode of each individual are initialized to obtain the best adaptive value of the group. According to the adaptive dynamic switching, the local search or global search is selected. The corresponding position's iterative formula is used to update the individual position, and the ERF model parameters are output to meet the iterative conditions. After obtaining the ERF model parameters, the ERF fault detection model based on the IBOA (IBOA-ERF) is constructed with the training data. The performance of the test model is tested by the real class labels of the test dataset and the predicted class labels generated by the model.
Start

Initializing butterfly population parameters, SCADA Datasets

SCADA Datasets

Data pre-processing

ERF Fault Detection Model

Obtaining parameters of ERF model

Obtaining the value of MAR, FAR

Calculate the fitness of each butterfly individual

Find the current optimal individual \( g^* \)

Choose a way of position iteration by the Equation (15) – (16)

\(|S_1(t)| > |S_2(t)|\)

Update the positions of current individual by the Equation (10)

Update the solution vector of the optimal position \( g^* \)

Update the sensory modality \( s \)

\( t = t + 1 \)

Ensure \( t \leq \text{MaxIters} \)

Update the positions of current individual by the Equation (14)

Ensure fitness(t) < fitness(t-1)

Current optimal solution \( g^* \)

ERF Fault Detection Model optimal parameters

end

Figure 7. Cont.
Figure 7. (a) The flow chart of IBOA to find the optimal parameters; (b) the flow chart of Data pre-process; (c) the flow chart of ERF Fault Detection Model.

Table 3 shows the optimized ERF hyperparameters $\tau$ and $\delta$ in the IBOA model, including the meanings and ranges of the parameters.

Algorithm 1 is the pseudo-code of the IBOA model’s parameters. Algorithm 2 is the pseudo-code of the ERF fault detection model using optimal parameters. The detailed optimization process of the model hyperparameters is as follows:

**Algorithm 1.** The steps of IBOA optimization parameters

Input: IBOA parameters ($lb(\tau_{min}, \delta_{min}), ub(\tau_{max}, \delta_{max})$); dimension: $dim$; maximum number of iterations: $MaxIter$; population size: $N$; ERF parameters ($\tau$, $\delta$);

Output: $g^*$ ($\tau_{optimal}$, $\delta_{optimal}$):

1: $x_{\text{train}}, y_{\text{train}}, x_{\text{test}}, y_{\text{test}} \rightarrow \text{ERF} (\tau, \delta)$
2: Initialize the butterfly population $N (i = 1, 2, \ldots, N)$
3: Calculate the fitness of each butterfly $g^*(\tau, \delta) = \text{the best individual}$
4: Build the fitness function: $fitness = FAR + \epsilon \times MAR$
5: While $t < MaxIter$
6: for $i = 1: N$
7: Calculate the perceived magnitude of the fragrance using Equation (4)
8: end for
9: Find the optimal butterfly individual $g^*$
10: for $i = 1: N$
11: if $|S_1(t)| > |S_2(t)|$
12: Enter the local search phase based on Equation (10)
13: else
14: Enter the global search phase based on Equation (14)
15: end if
16: end for
17: Check if each butterfly exceeds the search space and correct for it
18: Calculate the fitness of each butterfly
19: Select the location that matches the minimum fitness value
20: Update the value of $\alpha$
21: If a better solution is available, update $g^*$
22: $t = t + 1$
23: end while
24: return $g^*$ ($\tau_{optimal}$, $\delta_{optimal}$)
Algorithm 2. ERF Fault Detection Model

**Input:** the best parameter vector \( g^* (\tau_{\text{optimal}}, \delta_{\text{optimal}}) \); Training dataset; Test dataset;

**Output:** MAR, FAR
1: Training dataset \( \rightarrow x_{\text{train}}, y_{\text{train}} \)
2: Test dataset \( \rightarrow x_{\text{test}}, y_{\text{test}} \)
3: \( x_{\text{train}}, y_{\text{train}} \rightarrow \text{ERF} (\tau_{\text{optimal}}, \delta_{\text{optimal}}) \)
4: Training ERF fault detection model using Training dataset
5: \( x_{\text{test}}, y_{\text{test}} \rightarrow \text{ERF} (\tau_{\text{optimal}}, \delta_{\text{optimal}}) \)
6: Testing ERF fault detection model using Test dataset
7: Obtaining predicted labels of test datasets
8: Calculating MAR and FAR of model performance based on Equations (18) and (19)
9: \( \text{return} \) MAR, FAR

| Parameter | Meaning | Value Range |
|-----------|---------|-------------|
| \( n_{\text{estimators}} \) \((\tau)\) | The number of decision trees in ERF | [10, 1000] |
| \( \text{max\_depth} \) \((\delta)\) | Maximum depth of the decision tree | [10, 200] |

6. Experimental Analysis

6.1. Dataset Description

To validate the validation of the proposed IBOA-ERF fault detection model, the annual gearbox operation data were extracted from the SCADA dataset with an interval of 1 min for a 1.5 MW wind turbine in China, and the data structure was selected from 30 min before the occurrence of the gearbox fault to 30 min after the end of the fault through the analysis of the wind turbine structure, as shown in Table 4.

| Features | Time | 18:12 | 18:13 | 18:14 | ... | 19:40 | 19:41 | 19:42 |
|----------|------|-------|-------|-------|-----|-------|-------|-------|
| nacelle_temperature | -8.5 | -8.7 | -8.8 | ... | 9.3 | 9.5 | 9.8 |
| wind_speed_1 | 10.01 | 9.94 | 9.34 | ... | 5.92 | 6.12 | 6.01 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| hydraulic_main_sys_pressure | 135.87 | 136.18 | 135.26 | ... | 144.72 | 144.72 | 144.11 |
| hydraulic_rotor_brake_sys_pressure | 149.30 | 148.69 | 149.90 | ... | 170.36 | 170.05 | 170.05 |

For the purposes of the dataset, as illustrated in Table 5, the dataset can be divided into two parts: Dataset 1, with data on gearbox supercapacitor overtemperature faults and fault-free data; and Dataset 2, with data on gearbox nacelle operation overspeed faults and fault-free data.

| Dataset | Fault-Free | Faulty | Total Number of Samples | Total Number of Features |
|---------|------------|-------|-------------------------|-------------------------|
| Dataset 1 | 1059 | 991 | 2050 | 210 |
| Dataset 2 | 1211 | 1080 | 2291 | 210 |

6.2. Criteria for Evaluation

For the dichotomous problem of wind turbine gearbox fault detection, a confusion matrix was introduced. As illustrated in Table 6, the missing alarm rate (MAR) and the false alarm rate (FAR) of the matrix were utilized as evaluation indices.
Table 6. Confusion matrix for binary classification problems.

| Actual Category | Predict Category |
|-----------------|------------------|
| Normal          | \( S_{TN} \) (true negative) | \( S_{FP} \) (false positive) |
| Fault           | \( S_{FN} \) (false negative) | \( S_{TP} \) (true positive) |

\[
\text{MAR} = \frac{S_{FN}}{S_{FN} + S_{TP}} \quad (18)
\]

\[
\text{FAR} = \frac{S_{FP}}{S_{FP} + S_{TN}} \quad (19)
\]

where \( S_{FN} \), \( S_{FP} \), \( S_{TN} \), and \( S_{TP} \) represent the corresponding sample size.

To validate the excellence of ERF under the IBOA for the above extracted dataset, after data pre-processing, it was compared with the ERF model under MFO, MVO, SSA, SCA, and BOA optimization, and evaluated the performance of each model using MAR and FAR. Lower values of MAR and FAR represent better performance of the model. In order to prevent overfitting and improve model accuracy, each model was trained using 10-level cross-validation when conducting the comparison experiments. At the same population size and number of iterations, each model was run 10 times individually.

6.3. Experimental Results

When comparing the MAR and FAR of the ERF model under different optimization algorithms, IBOA-ERF performed better than the other five models.

For Dataset 1, as shown in Figure 8a, for the MAR of the six models, the average MAR of IBOA-ERF running 10 times alone was 0.86%, which is significantly improved compared with the BOA algorithm, and the fault detection ability is very stable. The overall MAR was maintained at 0.72–0.98%, while that of the other models was maintained at 0.84–1.53%. The optimization ability and optimization accuracy of the model were greatly improved. As shown in Figure 8b, for the FAR of the six models, the average FAR of IBOA-ERF running alone 10 times was 5.30%. During the detection process, the FAR of MFO-ERF was up to 9.23%, and the optimization effect was not obvious, while that of IBOA-ERF was maintained between 4.87% and 5.91%, and the detection performance was very stable. This shows that the ERF model has lower MAR and FAR, and the convergence efficiency and optimization performance are greatly improved when using the optimization parameters of the IBOA.

For Dataset 2, as shown in Figure 8c, the MAR of the ERF model under the IBOA had a maximum decrease of 1.06% compared to the other five models, showing less fluctuation than the classification results of the other models—which were generally maintained between 0.54% and 0.77%—along with significantly improved detection performance compared to the other models. As shown in Figure 8d, the FAR of the ERF model under the IBOA was generally stable between 4.97% and 6.65%, while the FAR of the other five models mostly remained above 6.13%, with the maximum reaching 9.75%. The IBOA has obvious optimization effects, is not prone to becoming trapped in partial optima, and shows greatly improved accuracy.
mode almost remained above 6.13%, with the maximum reaching 9.75%. The IBOA has obvious optimization effects, is not prone to be trapped in partial optima, and shows greatly improved accuracy.

Figure 8. (a) MAR of six algorithms for fault detection on Dataset 1; (b) FAR of six algorithms for fault detection on Dataset 1; (c) MAR of six algorithms for fault detection on Dataset 2; (d) FAR of six algorithms for fault detection on Dataset 2.

7. Conclusions

Aiming at the difficulty of parameter optimization of wind turbine gearbox fault detection models, the IBOA-ERF fault detection model was proposed. The IBOA was used to optimize the hyperparameters of ERF, so as to improve the detection performance.

There are four main contributions of this paper: First, chaotic mapping is introduced to replace the original population initialization method to enhance the randomness of the population distribution and enhance the local development and global exploration capabilities. Second, the adaptive inertia weight factor is designed and combined with the landmark operator of PIO, so that the best position information of individual history is more effectively used, and it is integrated into the position update formula to improve the diversity and robustness of the BOA. Third, a new dynamic switching method of the search stage is designed, so that two search phases can reach a dynamic balance, preventing a drop into local optima and accelerating convergence. Finally, an improved fault detection model for wind turbine gearboxes is proposed by combining the above strategies with ERF.

In the experiments, MFO, MVO, SSA, SCA, BOA, and IBOA were introduced to enhance experimental fairness, each used to act on the ERF model, and the fitness function was constructed. MAR and FAR were used as assessment indicators. The results indicate that when using the IBOA to optimize the ERF parameters, the MAR and FAR are still low when the dataset is complex and the dimensionality is high.

Based on the proposed IBOA-ERF wind turbine gearbox fault detection model, the recommendations for future research are as follows:
• When the data categories are unbalanced—that is, when there are many normal samples and few fault samples—further research can be conducted to solve the problem of the model detection being biased towards the majority of samples, and the classification accuracy is reduced.

• With the upgrading of the wind turbine gearbox technology, the feature dimensionality and complexity of the original dataset can increase. There are many data preprocessing methods and no uniform measurement, which can influence the implementation of the model. The data pre-processing methods that are most suitable for this model can be further studied.

• The IBOA can be applied to other fault detection fields.

Author Contributions: Supervision, M.T.; writing—original draft, C.C.; formal analysis, H.W.; investigation, H.Z.; data curation, J.T.; conceptualization, Z.P.; visualization, Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China (grant no. 62173050), the National Key R&D Program of China (grant no. 2019YFE0105300), the Energy Conservation and Emission Reduction Hunan University Student Innovation and Entrepreneurship Education Center, Changsha University of Science and Technology’s “The Double First Class University Plan” International Cooperation and Development Project in Scientific Research in 2018 (Grant No. 2018C14), the Hunan Provincial Department of Transportation’s 2018 Science and Technology Progress and Innovation Plan Project (grant no. 201843), the Hubei Superior and Distinctive Discipline Group of “New Energy Vehicle and Smart Transportation”, the Open Fund of Hubei Key Laboratory of Power System Design and Test for Electrical Vehicle (grant no. ZDSYS202201), General Projects of Hunan University Students’ Innovation and Entrepreneurship Training Program in 2022 (grant no. 2565), and the Graduate Scientific Research Innovation Project of Changsha University of Science and Technology (grant no. CXCLY2022094).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

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

Nomenclature and Abbreviations

| Abbreviation | Description |
|--------------|-------------|
| GWEC         | Global Wind Energy Council |
| SCADA        | Supervisory control and data acquisition |
| NFSW-BP-AdaBoost | Non-fuzzy solution-weighted BP-AdaBoost |
| XGBoost      | Extreme gradient boosting |
| CS-GBDT      | Cost-sensitive GBDT |
| RF           | Random forest |
| ERF          | Extreme random forest |
| BOA          | Butterfly optimization algorithm |
| IBOA         | Improved butterfly optimization algorithm |
| IBOA-ERF     | Extreme random forest optimized by improved butterfly optimization algorithm |
| PIO          | Pigeon-inspired optimization |
| MFO          | Moth–flame optimization |
| MVO          | Multi-verse optimization |
| SCA          | Sine–cosine algorithm |
| SSA          | Salp swarm algorithm |
| UCI          | University of California Irvine |
| FAR          | False alarm rate |
| MAR          | Missing alarm rate |
Symbols

| Symbol | Description |
|--------|-------------|
| D      | Number of numbers in ERF |
| \( P_t(c|v_i) \) | Probability that the sample belongs to category c conditional on the feature vector \( v_i \) |
| \( v_i \) | Feature vector of the sample |
| \( \bar{P}(c|v_i) \) | Average value of \( P_t(c|v_i) \) in the ERF |
| c      | Some kind of category |
| \( \hat{c} \) | Category corresponding to the maximum value of \( P(c|f_i) \) |
| \( I_k \) | Mutual information of the two subsets |
| \( H_k \) | Split entropy of the feature k |
| \( H_c \) | Information entropy |
| \( \text{Score}_k \) | Score measurement of the calculated feature |
| f      | Fragrance intensity |
| I      | Stimulus intensity |
| s      | Sensory modality |
| \( \alpha \) | Power exponent |
| \( x^t_i \) | Location information of the i-th individual in the t-th iteration |
| \( x^{t+1}_i \) | Location information of the i-th individual in the t + 1-th iteration |
| r      | Random value from 0 to 1 |
| \( g^* \) | Best value in the current iteration |
| \( f_i \) | Fragrance intensity emitted by the i-th individual |
| j      | Random number |
| k      | Random number |
| X      | Position parameter |
| t      | Current number of iterations |
| \( \mu \) | Logistics parameter |
| \( T_{\text{iter}} \) | Largest value of the number of iterations t |
| \( \omega \) | Adaptive inertia weight |
| e      | Euler number |
| \( x^t_c \) | Position of the center of the flock in the current iteration |
| \( \text{Fit}(x^t_i) \) | Value of the fitness function of the i-th individual |
| N      | Number of individuals |
| \( \emptyset \) | 100*\( \pi \) |
| \( \text{cov} \) | Covariance |
| \( \sigma \) | Standard deviation |
| \( \rho \) | Correlation coefficient |
| S      | Corresponding sample size |

References

1. GWEC. Global Wind Energy Council (GWEC) | Global Wind Report. 2022. Available online: https://gwec.net/global-wind-report-2022/ (accessed on 4 April 2022).
2. Han, Z.; Liu, Z.; Kang, W.; He, W. Boundary Feedback Control of a Nonhomogeneous Wind Turbine Tower with Exogenous Disturbances. IEEE Trans. Autom. Control 2022, 67, 1952–1959. [CrossRef]
3. Liu, Z.; Zhang, L. Zhang. A review of failure modes, condition monitoring and fault diagnosis methods for large-scale wind turbine bearings. Measurement 2020, 149, 107002. [CrossRef]
4. Jiang, G.; He, H.; Yan, J.; Xie, P. Multiscale Convolutional Neural Networks for Fault Diagnosis of Wind Turbine Gearbox. IEEE Trans. Ind. Electron. 2019, 66, 3196–3207. [CrossRef]
5. Qin, Y.; Wang, X.; Zou, J. The Optimized Deep Belief Networks with Improved Logistic Sigmoid Units and Their Application in Fault Diagnosis for Planetary Gearboxes of Wind Turbines. IEEE Trans. Ind. Electron. 2019, 66, 3814–3824. [CrossRef]
6. Tang, M.; Zhao, Q.; Ding, S.X.; Wu, H.; Li, L.; Long, W.; Huang, B. An Improved LightGBM Algorithm for Online Fault Detection of Wind Turbine Gearboxes. Energies 2020, 13, 807. [CrossRef]
7. Tang, M.; Yi, J.; Wu, H.; Wang, Z. Fault Detection of Wind Turbine Electric Pitch System Based on IGWO-ERF. Sensors 2021, 21, 6215. [CrossRef]
8. Tang, M.; Zhao, Q.; Wu, H.; Wang, Z. Cost-Sensitive LightGBM-Based Online Fault Detection Method for Wind Turbine Gearboxes. Front. Energy Res. 2021, 9, 378. [CrossRef]
9. Wang, D.; Zhao, Y.; Yi, C.; Tsui, K.L.; Lin, J. Sparsity guided empirical wavelet transform for fault diagnosis of rolling element bearings. Mech. Syst. Signal Process. 2018, 101, 292–308. [CrossRef]
10. Chen, H.; Jiang, B.; Ding, S.X.; Huang, B. Huang. Data-Driven Fault Diagnosis for Traction Systems in High-Speed Trains: A Survey, Challenges, and Perspectives. *IEEE Trans. Intel1. Transp. Syst.* 2022, 23, 1700–1716. [CrossRef]

11. Wang, Y.; Pan, Z.; Yuan, X.; Yang, C.; Gui, W. A novel deep learning based fault diagnosis approach for chemical process with extended deep belief network. *ISA Trans.* 2020, 96, 457–467. [CrossRef]

12. Liang, J.; Qin, Z.; Xiao, S.; Ou, L.; Lin, X. Efficient and Secure Decision Tree Classification for Cloud-Assisted Online Diagnosis Services. *IEEE Trans. Dependable Secur. Comput.* 2021, 18, 1632–1644. [CrossRef]

13. Stetco, A.; Dimohammadi, F.; Zhao, X.; Robu, V.; Flynn, D.; Barnes, M.; Nenadic, G. Machine learning methods for wind turbine condition monitoring: A review. *Renew. Energy* 2019, 133, 620–635. [CrossRef]

14. Feng, D.C.; Liu, Z.T.; Wang, X.D.; Chen, Y.; Chang, J.Q.; Wei, D.F.; Jiang, Z.M. Machine learning-based compressive strength prediction for concrete: An adaptive boosting approach. *Constr. Build. Mater.* 2020, 230, 117000. [CrossRef]

15. Jiang, H.W.; Zou, B.; Xu, C.; Xu, J.; Tang, Y.Y. SVM-Boosting based on Markov resampling: Theory and algorithm. *Neural Netw.* 2016, 131, 276–290. [CrossRef] [PubMed]

16. Liu, Y.; Zhao, C.C.; Liang, H.L.; Lu, H.H.; Cui, N.Y.; Bao, K.Y. A rotor fault diagnosis method based on BP-Adaboost weighted by non-fuzzy solution coefficients. *Measurement* 2022, 196, 111280. [CrossRef]

17. Chakraborty, D.; Elzarka, H. Early detection of faults in HVAC systems using an XGBoost model with a dynamic threshold. *Energy Build.* 2019, 185, 326–344. [CrossRef]

18. Xu, Q.F.; Lu, S.X.; Jia, W.Y.; Jiang, C.X. Imbalanced fault diagnosis of rotating machinery via multi-domain feature extraction and cost-sensitive learning. *J. Intell. Manuf.* 2020, 31, 1467–1481. [CrossRef]

19. Yang, L.; Shami, A. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing* 2020, 415, 295–316. [CrossRef]

20. Yang, X.S. Nature-inspired optimization algorithms: Challenges and open problems. *Adv. Eng. Softw.* 2022, 185, 103616. [CrossRef]

21. Luo, J.; Tian, Q.; Xu, M. Reverse guidance butterfly optimization algorithm integrated with information cross-sharing. *J. Intell. Fuzzy Syst.* 2021, 41, 3463–3484. [CrossRef]

22. Neshat, M.; Nezhad, M.M.; Abbasnejad, E.; Mirjalili, S.; Tjernberg, L.B.; Garcia, D.A.; Alexander, B.; Wagner, M. A deep learning-based evolutionary model for short-term wind speed forecasting: A case study of the Lillgrund offshore wind farm. *Energy Convers. Manag.* 2021, 236, 114002. [CrossRef]

23. Song, D.R.; Li, Z.Q.; Wang, L.; Jin, F.J.; Huang, C.E.; Xia, E.; Rizk-Allah, R.M.; Yang, J.; Su, M.; Joo, Y.H. Energy capture efficiency enhancement of wind turbines via stochastic model predictive yaw control based on intelligent scenarios generation. *Appl Energy* 2022, 312, 118773. [CrossRef]

24. Song, D.R.; Tu, Y.P.; Wang, L.; Jin, F.J.; Li, Z.Q.; Huang, C.N.; Xia, E.; Rizk-Allah, R.M.; Yang, J.; Su, M.; et al. Coordinated optimization on energy capture and torque fluctuation of wind turbines via variable weight NMPC with fuzzy regulator. *Appl. Energy* 2022, 312, 118821. [CrossRef]

25. Azamfar, M.; Singh, J.; Bravo-Imaz, I.; Lee, J. Multisensor data fusion for gearbox fault diagnosis using 2-D convolutional neural network and motor current signature analysis. *Mech. Syst. Signal Process.* 2020, 144, 106861. [CrossRef]

26. Long, W.; Jiao, J.J.; Liang, X.M.; Wu, T.B.; Xu, M.; Cai, S.H. Pinhole-imaging-based butterfly optimization algorithm for global optimization and feature selection. *Appl. Soft Comput.* 2021, 103, 107146. [CrossRef]

27. Long, W.; Wu, T.B.; Xu, M.; Tang, M.Z.; Cai, S.H. Parameters identification of photovoltaic models by using an enhanced adaptive butterfly optimization algorithm. *Energy* 2021, 229, 120750. [CrossRef]

28. Song, D.R.; Tu, Y.P.; Wang, L.; Jin, F.J.; Li, Z.Q.; Huang, C.N.; Xia, E.; Rizk-Allah, R.M.; Yang, J.; Su, M.; et al. Coordinated optimization on energy capture and torque fluctuation of wind turbines via variable weight NMPC with fuzzy regulator. *Appl. Energy* 2022, 312, 118821. [CrossRef]

29. Zhang, Y.Q.; Wang, X.Y. A symmetric image encryption algorithm based on mixed linear-nonlinear coupled map lattice. *Inf. Sci.* 2014, 273, 329–351. [CrossRef]

30. Hua, Z.Y.; Zhou, Y.C. Exponential Chaotic Model for Generating Robust Chaos. *IEEE Trans. Syst. Man Cybern.-Syst.* 2021, 51, 3713–3724. [CrossRef]

31. Duan, H.B.; Wang, X.H. Echo State Networks with Orthogonal Pigeon- Inspired Optimization for Image Restoration. *IEEE Trans. Neural Netw. Learn. Syst.* 2016, 27, 2413–2425. [CrossRef]

32. Cui, Z.H.; Zhang, J.J.; Wang, Y.C.; Cao, Y.; Cai, X.; Zhang, W.J.; Chen, J.J. A pigeon-inspired optimization algorithm for many-objective optimization problems. *Sci. China-Inf. Sci.* 2019, 62, 70212. [CrossRef]

33. Shehab, M.; Abualigah, L.; Al Hamad, H.; Alabool, H.; Alshinwan, M.; Khasawneh, A.M. Moth-flame optimization algorithm: Variants and applications. *Neural Comput. Appl.* 2020, 32, 9859–9884. [CrossRef]

34. Mirjalili, S.; Jatamli, M.; Mirjalili, S.M.; Hatamlou, A. Multi-Verse Optimizer: A nature-inspired algorithm for global optimization. *Neural Comput. Appl.* 2016, 27, 495–513. [CrossRef]

35. Abualigah, L.; Diabat, A. Advances in Sine Cosine Algorithm: A comprehensive survey. *Artif. Intell. Rev.* 2021, 54, 2567–2608. [CrossRef]

36. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* 2017, 114, 163–191. [CrossRef]
38. Langfelder, P.; Horvath, S. Fast R Functions for Robust Correlations and Hierarchical Clustering. *J. Stat. Softw.* 2012, *46*, 1–17. [CrossRef]

39. Zhang, K.; Peng, K.X.; Ding, S.X.; Chen, Z.W.; Yang, X. A Correlation-Based Distributed Fault Detection Method and Its Application to a Hot Tandem Rolling Mill Process. *IEEE Trans. Ind. Electron.* 2020, *67*, 2380–2390. [CrossRef]