


## Article

# A Fault Warning Approach Using an Enhanced Sand Cat Swarm Optimization Algorithm and a Generalized Neural Network

Youchun Pi <sup>1</sup>, Yun Tan <sup>1,\*</sup>, Amir-Mohammad Golmohammadi <sup>2,\*</sup> , Yujing Guo <sup>1</sup>, Yanfeng Xiao <sup>1</sup> and Yan Chen <sup>3</sup><sup>1</sup> China Yangtze Power, No. 1 Xiba Construction Road, Xiling District, Yichang 443000, China<sup>2</sup> Department of Industrial Engineering, Faculty of Engineering, Arak University, 3848177584 Arak, Iran<sup>3</sup> Independent Researcher, 201, Building 7, Baolong Plaza, Lane 2449 Jinhai Road, Pudong New Area, Shanghai 201209, China

\* Correspondence: tan\_yun@ctg.com.cn (Y.T.); a-golmohammadi@araku.ac.ir (A.-M.G.)

**Abstract:** With the continuous development and complexity of industrial systems, various types of industrial equipment and systems face increasing risks of failure during operation. Important to these systems is fault warning technology, which can timely detect anomalies before failures and take corresponding preventive measures, thereby reducing production interruptions and maintenance costs, improving production efficiency, and enhancing equipment reliability. Machine learning techniques have proven highly effective for fault detection in modern production processes. Among numerous machine learning algorithms, the generalized neural network stands out due to its simplicity, effectiveness, and applicability to various fault warning scenarios. However, the increasing complexity of systems and equipment presents significant challenges to the generalized neural network. In real-world scenarios, it suffers from drawbacks such as difficulties in determining parameters and getting trapped in local optima, which affect its ability to meet the requirements of high efficiency and accuracy. To overcome these issues, this paper proposes a fault warning method based on an enhanced sand cat swarm optimization algorithm combined with a generalized neural network. First, we develop an enhanced sand cat swarm optimization algorithm that incorporates an improved chaotic mapping initialization strategy, as well as Cauchy mutation and reverse elite strategies based on adaptive selection. Subsequently, we utilize this algorithm to optimize the generalized neural network and determine its optimal parameters, effectively improving the accuracy and reliability of system fault warnings. The proposed method is validated using actual industrial system data, specifically for generator fault warning, and is demonstrated to outperform other advanced fault warning techniques. This research provides valuable insights and promising directions for enhancing industrial fault warning capabilities.

**Keywords:** fault detection; generalized neural network; industrial systems; machine learning; sand cat swarm optimization



**Citation:** Pi, Y.; Tan, Y.; Golmohammadi, A.-M.; Guo, Y.; Xiao, Y.; Chen, Y. A Fault Warning Approach Using an Enhanced Sand Cat Swarm Optimization Algorithm and a Generalized Neural Network. *Processes* **2023**, *11*, 2543. <https://doi.org/10.3390/pr11092543>

Academic Editors: Amir M. Fathollahi-Fard, Vigen H. Arakelian, Zhiwu Li, Zixian Zhang, Guangdong Tian and Wei Sun

Received: 3 August 2023

Revised: 12 August 2023

Accepted: 14 August 2023

Published: 25 August 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

In today's constantly evolving and complex industrial systems, various industrial equipment and systems face increasing risks of failure, posing significant challenges to production and maintenance work [1]. Therefore, fault warning technology has become an immensely critical research area in the industrial sector; fault warning involves monitoring and analyzing data from systems, equipment, or processes to promptly identify potential faults or anomalies and issue advance warnings. The goal of fault warning is to detect early signs of potential issues, thereby reducing production disruptions, equipment damage, and maintenance costs and enhancing system reliability and safety. It typically entails collecting, monitoring, and analyzing data related to system operation, such as temperature, vibration, current, pressure, and other parameters. By comparing real-time data with predefined thresholds or models, the system can identify deviations from normal operation and alert

relevant personnel or systems for intervention through alarms, notifications, or automated controls. In industries such as manufacturing, energy, transportation, healthcare, and more, fault warning holds significant importance. It helps to improve equipment utilization, reduce maintenance costs, mitigate accident risks, and provide more reliable and continuous operation [2]. Consequently, developing effective early warning and diagnostic methods for equipment and system faults has become an urgent problem to address [3]. In this context, adopting machine learning-based fault prediction methods has emerged as a crucial approach to modern industrial fault prevention.

Neural network technology has been widely applied in the field of fault warning due to its powerful learning capability, rapid information processing speed, and error self-adaptation advantages [3]. Compared to other neural networks, the generalized neural network (GRNN) exhibits fewer parameters (only the smoothing factor  $\sigma$ ) and possesses strengths such as strong data adaptability, efficiency, and rapid prediction capabilities [4]. However, the performance of GRNN largely depends on the appropriate selection of  $\sigma$ , and determining its optimal parameters poses a challenge, as it may be difficult to find global optima in certain cases and can easily get trapped in local optima [5]. To better address these issues, intelligent algorithms have become essential tools [6].

Meanwhile, the No Free Lunch theorem states that no single algorithm can achieve optimal solutions for all problems [7]. Therefore, there is a continuous need to develop and improve novel algorithms and extend their applications. In this regard, this paper introduces an enhanced sand cat swarm optimization (ESCSO), specifically utilizing chaotic mapping for population initialization and incorporating Cauchy mutation and elite reverse population strategies based on adaptive selection during the standard sand cat swarm optimization (SCSO) search process. Subsequently, the algorithm is employed to optimize the GRNN by adjusting its parameter values to better adapt to the characteristics of complex systems. Through this approach, the network is better equipped to fit the nonlinearity and time-varying nature of complex systems.

Real industrial system data is used for experimentation to validate the proposed method's effectiveness. The results demonstrate significant performance advantages of this method in fault prediction, with higher accuracy and reliability compared to traditional approaches. This offers novel insights and methods for fault warning research in the industrial domain and provides robust support for accurate and reliable fault warnings in industrial production, promising to contribute positively to the robust operation of industrial systems and increased production efficiency.

In conclusion, the fault warning method proposed in this paper represents a promising research direction, with the potential to provide more effective solutions for fault prediction in the industrial sector while offering an intriguing new domain for academic exploration. Through continuous optimization and improvement, this method is expected to become a crucial technical means for future industrial system fault warnings. As technology advances and data accumulates, we firmly believe that this method will play an increasingly significant role in the industrial field, making greater contributions to the robust operation and increased production efficiency of industrial systems. In addition, this is the first combination of SCSO and GRNN in the field of fault warning, which is another contribution of this paper.

The remaining sections of this paper are organized as follows: Section 2 provides a comprehensive review of recent work related to fault warning, Section 3 elaborates on our proposed method, Section 4 validates the method through a real-world case study on generator fault warning, Section 5 further demonstrates the effectiveness of the proposed method through comparisons with other advanced techniques, and finally, Section 6 presents the conclusion, limitations, and potential directions for future development.

## 2. Literature Review

Many methods have emerged in the field of fault warning in the past decades. For instance, Wu et al. formulated a combined approach that integrated a deep local adap-

tive network, dual-phase qualitative trend analysis, and a five-state Bayesian network. This method turned aberrant variable continuous data into trend state data, serving fault detection, identification, and diagnosis [8]. Luo et al. introduced an online monitoring technology based on SCADA data to enhance prediction reliability. They used the conditional mutual information method to select key variables, which improved prediction accuracy. Their innovative Pair-Copula model overcame the limitations of traditional models, simplifying prediction complexity. By combining the back propagation neural network (BPNN) with the Pair-Copula model, they enhanced prediction effectiveness. The improved Pair-Copula model addressed real-time data processing challenges and demonstrated practicality, achieving significant progress in the field of wind turbine maintenance [9]. Compared to traditional methods, Chen et al. utilized mixed intelligent algorithms to handle mechanical fault diagnosis more effectively. They employed Parallel factor analysis (PARAFAC) for multi-dimensional signal processing and constructed a 3D tensor using continuous wavelet transform to extract complex features efficiently. By applying genetic algorithms for optimization in the diagnosis of centrifugal pump faults, they further improved accuracy and significantly enhanced the efficiency of fault diagnosis [10]. Jiang et al. proposed a condition-based maintenance method for silk dryers, using entropy methods to enhance objectivity. They improved prediction accuracy by optimizing neural network nodes and adopted genetic algorithm-backward propagation neural network (GA-BP) to establish a state prediction model, overcoming limitations of BP networks. Simulation experiments validated the effectiveness of their method in compensating for existing maintenance deficiencies and providing a scientific basis for dryer maintenance [11]. Zhao et al. introduced the use of deep autoencoder networks for wind turbine health monitoring, enabling fault detection and analysis using SCADA data. Their method not only achieved early fault warnings but also inferred the fault component's location, distinguishing it from previous research [12]. Chen et al. improved the accuracy of wind turbine pitch system and fault early warnings by optimizing the BP neural network using a GA. Although the integration of multiple methods resulted in relatively higher time and space complexity, the early warning capability reduced subsequent maintenance costs [13]. In comparison to previous research, Zhang et al. used an improved gray wolf optimization algorithm (IGWO) and the BP neural network to enhance accuracy and real-time capability in charging safety early warnings for electric vehicles. The IGWO-BP algorithm fitted the entire charging process, creating a warning model based on residual changes, reliably detecting abnormal states and issuing timely alerts, effectively enhancing electric vehicle charging safety warning capabilities [14]. Lin et al. upgraded a BPNN using an advanced sparrow search algorithm (SSA), significantly improving its performance. The fault classification and prediction accuracy for active phase change control device increased from 53.33% to 96.66%, validating the practicality and effectiveness of the algorithm, which holds significant significance for stable operation of stratospheric lighter-than-air aircraft [15]. Gao et al. designed an early warning system for EV charging processes using adaptive deep belief networks and charging data analysis. Experimental results demonstrated that their method not only accurately predicted faults during EV charging but also outperformed the BPNN and traditional deep belief network methods in early warning accuracy [16]. Zhou et al. innovatively proposed an entropy-based sparsity measurement method for bearing fault prediction and defect identification in complex hydraulic machinery. This method met important conditions and proved suitable for bearing degradation monitoring and envelope analysis. It surpassed existing methods in terms of algorithm performance and effectiveness, providing an effective tool for fault monitoring and diagnosis, albeit with slightly higher computational complexity [17]. Wang et al. introduced a multi-parameter signal-based early warning method for compressor valve faults, employing an improved deep learning network and information fusion strategy. Compared to previous research, their method achieved higher parameter prediction accuracy and reduced model complexity through decomposition, denoising, and reconstruction. Experiments verified its effectiveness in early warning compressor valve faults under complex

operating conditions [18]. Liu et al. presented a real-time detection method for working motor faults, utilizing vibration signal decomposition and feature computation for fault diagnosis and classification using support vector machines and neural networks. Their unique experimental process combined with hardware configuration enabled real-time motor fault diagnosis and the design of an early warning system [19]. Ji et al. introduced a multi-feature sparrow search algorithm (MSSA) optimized support vector machine (SVM) for fault early warning in cable tunnel environment monitoring, achieving online monitoring and intelligent alarms. Compared to traditional methods, their approach significantly improved accuracy [20]. Yang et al. proposed a method based on sparse autoencoder-long short-term memory (LSTM) network and sliding window technique for early warning of overheating defects in water-cooled turbine generator stator windings. By reconstructing operational data, LSTM temperature prediction, and defect detection through sliding windows, their approach provided more accurate prediction results compared to traditional methods, supporting early stable warnings [21]. Peng et al. combined convolutional neural network (CNN) with adaptive maximum mean discrepancy to develop a wind turbine fault early warning method. Industrial case studies demonstrated that their evaluation method had higher fault prediction rates and lower false alarm rates compared to traditional methods [22]. Kirbaş et al. designed a hybrid anomaly detection model combining multiple linear regression, response surface methods, and a multilayer perceptron for detecting faults in high-power generators [23]. Qiao et al. developed a meta-learning-based CNN approach for wind turbine fault early warning. Experimental results showed that their proposed method could detect faults in different wind turbine units at an early stage [24]. Li et al. optimized the CNN using an improved hybrid particle swarm optimization algorithm and applied it to fault early warning for synchronous generators. Their approach had significance in promoting stable operation of synchronous generators [25]. Niu et al. proposed a least squares hybrid support vector machine method for fault early warning in doubly fed wind turbine units. Research results indicated that their prediction method had higher estimation accuracy and could timely identify faults during the operation of doubly fed wind turbines [26]. Li et al. presented an intelligent early warning method for mine safety based on GRNN. Their method enhanced overall stability prediction for mines and effectively controlled errors to avoid vague predictions [27]. Kaminski et al. applied GRNN to rotor fault monitoring of induction motors, and real-world industrial cases demonstrated the effectiveness of GRNN in this domain [28]. Chen et al. combined principal component analysis with GRNN to establish a safety assessment and prediction model for coal mills. Experimental cases showed that their model had lower costs and higher accuracy [29]. Liu et al. developed a wind turbine performance prediction model using GRNN and implemented fault early warning using a sliding data window method to calculate residual evaluation indicators for wind turbine speed and power in real time. Experimental cases showed that their model outperformed traditional methods [30]. Qi et al. combined kernel entropy component analysis (KECA) and GRNN for health monitoring and fault early warning of wind turbines. Observed results from their model indicated that it could provide early safety warnings for faults [31]. Jing et al. developed a hybrid method combining KECA and GRNN for gear box fault monitoring and early warning in wind turbine units. Their method improved fault warning efficiency by systematically integrating dimensionality reduction techniques and GRNN [32].

Finally, Table 1 summarizes the above literature.

**Table 1.** Summary of Literature review.

| Ref.             | Methodology                    | Application                | Key Findings/Contributions                                |
|------------------|--------------------------------|----------------------------|---|
| Luo et al. [9]   | Conditional mutual info + BPNN | Wind turbine maintenance   | Enhanced reliability with conditional mutual information. |
| Chen et al. [10] | Parallel factor decomp + GA    | Mechanical fault diagnosis | Improved efficiency using PARAFAC and genetic algorithms. |

Table 1. Cont.

| Ref.                 | Methodology   | Application  | Key Findings/Contributions                                       |
|----------------------|---|--|--|
| Jiang et al. [11]    | Genetic algorithm + BPNN                              | Silk dryer maintenance   | Optimized neural network nodes for improved predictions.         |
| Zhao et al. [12]     | Deep autoencoder network                              | Wind turbine health monitoring   | Early fault warnings and accurate fault location identification. |
| Chen et al. [13]     | Genetic algorithm + BPNN                              | Wind turbine pitch system fault warning                                    | Enhanced early warning capability.                               |
| Zhang et al. [14]    | IWOA + BPNN   | EV charging safety early warnings  | Reliable and real-time abnormal state detection.                 |
| Lin et al. [15]      | SSA+ BPNN   | Stratospheric lighter-than-air aircraft                                    | Significant improvement in fault classification accuracy.        |
| Gao et al. [16]      | Adaptive deep belief networks + Charging data         | EV charging process  | Accurate fault prediction, outperforming traditional methods.    |
| Zhou et al. [17]     | Entropy-based sparsity + LSTM network                 | Hydraulic machinery bearing fault pred.                                    | Effective tool for fault monitoring and diagnosis.               |
| Wang et al. [18]     | Multi-stage fusion LSTM network                       | Compressor valve faults  | Higher parameter prediction accuracy with reduced complexity.    |
| Liu et al. [19]      | Support vector machines + GRNN                        | Motor fault detection  | Real-time motor fault diagnosis and early warning system.        |
| Ji et al. [20]       | MSSA+ optimized SVM                                   | Cable Tunnel Environment Monitoring  | Significant accuracy improvement for generator fault warning     |
| Yang et al. [21]     | Sparse AE+ LSTM+ Sliding Window                       | Water-Cooled Steam Turbine Generator Stator Winding Overheating Warning    | More accurate early-stage stable warnings                        |
| Peng et al. [22]     | Convolutional Network+ Adaptive Max Mean Deviation    | Wind Turbine Generator Fault Warning                                       | Higher fault prediction, lower false alarms                      |
| Kirbaş et al. [23]   | Multivariate Linear Regression+ Response Surface+ MLP | High-Power Generator Fault Detection                                       | Hybrid model for improved fault detection                        |
| Qiao et al. [24]     | Meta-Learning+ CNN                                    | Wind Turbine Generator Fault Warning                                       | Early fault detection across different units                     |
| Li et al. [25]       | Improved Hybrid PSO+ CNN                              | Synchronous Generator Fault Warning  | Enhanced stability through optimized CNN model                   |
| Niu et al. [26]      | Least Squares Hybrid SVM                              | Doubly-Fed Wind Turbine Group Fault Warning                                | Accurate fault estimation, timely identification                 |
| Li et al. [27]       | GRNN-based Intelligent Warning for Mines              | Overall Stability Forecast for Mines                                       | Improved mine stability prediction with intelligent warning      |
| Kaminski et al. [28] | GRNN  | Motor Rotor Fault Monitoring   | Effective motor rotor fault monitoring                           |
| Chen et al. [29]     | PCA-GRNN  | Coal Mill Safety Assessment Prediction                                     | Lower cost, higher accuracy in coal mill safety prediction       |
| Liu et al. [30]      | GRNN  | Wind Turbine Group Performance Prediction and Sliding Window Fault Warning | Superior performance in predicting turbine group performance     |
| Qi et al. [31]       | KECA+ GRNN  | Wind Turbine Health Monitoring and Fault Warning                           | Early warnings for turbine health and faults using KECA and GRNN |
| Jing et al. [32]     | KECA + GRNN   | Wind Turbine Gearbox Fault Monitoring and Warning                          | Enhanced fault warning through hybrid KECA and GRNN              |

The analysis of the literature (Table 1) reveals that machine learning techniques are widely applied in the field of fault warning. However, the utilization of GRNN in current effective methods is somewhat limited. Table 1 presents a clear overview of GRNN applications, with both the basic version and hybrid combinations with other methods being more prevalent. This also implies the existence of unexplored opportunities to enhance performance and broaden its application. The primary goal of this study is to propose a more streamlined and efficient approach for utilizing GRNN. This involves optimizing the performance of the basic version to improve efficiency and fully leverage its simplicity. Additionally, based on the findings in Table 1, meta-heuristic algorithms prove effective in

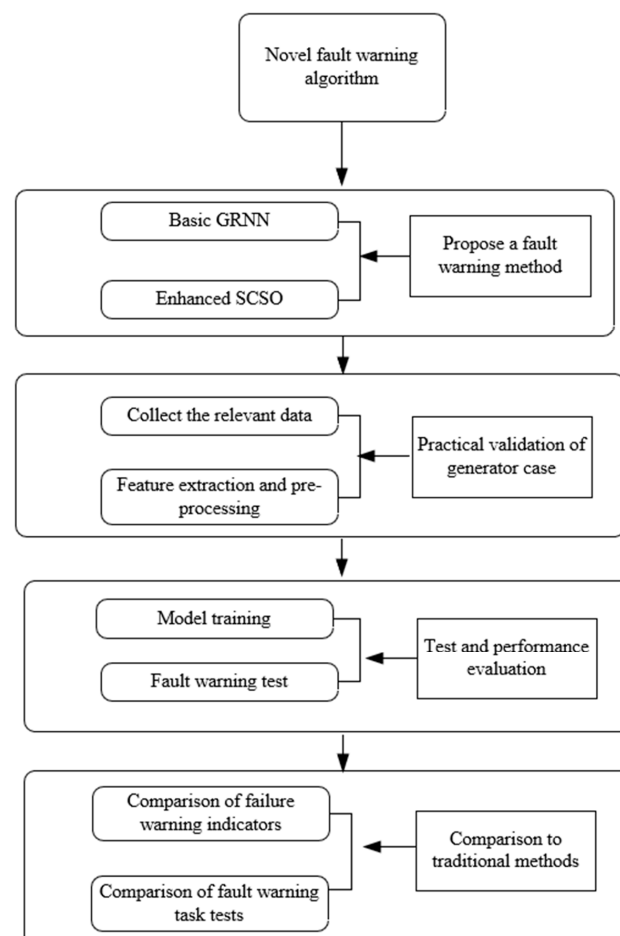


enhancing machine learning performance. By selecting suitable meta-heuristics for specific problems, a significant performance boost for GRNN across various tasks is expected. It's important to note the "no free lunch" theorem, which asserts that no single algorithm can achieve optimal performance for all problems. This realization drives continuous innovation and algorithm refinement to meet evolving challenges. Therefore, another aim of this research is to explore an efficient meta-heuristic algorithm and integrate it with GRNN seamlessly. This combined approach aims to unlock improved and efficient utilization of GRNN, pushing the boundaries of fault warning. Through these efforts, our goal is to advance the field of fault warning and further explore the potential of GRNN in practical applications.

Hence, in comparison to prior research, this study makes unique contributions to the field:

- Building upon the foundational version of SCSO, we incorporate novel and improved operators to enhance the performance of SCSO.
- Drawing insights from the analysis of existing literature, meta-heuristic algorithms emerge as pivotal tools in optimizing machine learning efficiency. As a result, we apply ESCSO to GRNN to enhance the efficiency of fault diagnosis. This represents the first-ever combination of ESCSO and GRNN, and its effectiveness is validated through real-world industrial cases.
- In comparison to preceding research endeavors, we present a novel approach to optimal parameter calibration. This approach involves the synergy of the K-fold cross-validation technique and Taguchi's experimental method, underpinned by the concept of relative percentage deviation.

Finally, Figure 1 provides the lineage of our research throughout this paper.



**Figure 1.** Research lineage.

### 3. Proposed Hybrid Method

Here, we present the hybrid algorithm proposed in this paper, we first describe the GRNN structure (Section 3.1), after that, we detail the main components of ESCSO and provide pseudo-code for these cores (Section 3.2), and finally, we describe ESCSO-GRNN and provide its flowchart (Section 3.3).

#### 3.1. Generalized Neural Network Structure

Specht proposed the GRNN model [33]. This model is typically built on the foundation of mathematical statistics and exhibits powerful capabilities in non-linear mapping and fast learning. The GRNN neural network model consists of input layer, pattern layer, summation layer, and output layer [33]. Figure 2 illustrates the structure of a GRNN.

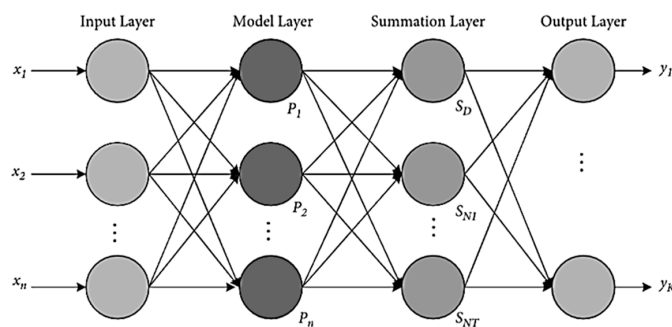


Figure 2. GRNN structure.

In the GRNN, the input layer receives the feature vector of input samples, and the pattern layer is a crucial component, storing the feature vectors of training samples along with their corresponding output values. The summation layer calculates the weights by computing the similarity between the input sample and the samples in the pattern layer and then performs a weighted summation of the output values. Finally, the output layer uses the result of the weighted summation as the final prediction output.

The advantages of GRNN lie in its minimal number of parameters, with only one smoothing factor  $\sigma$ , requiring less parameter tuning. Since the pattern layer stores all the training samples, GRNN possesses a powerful memory capability, allowing it to consider information from all known samples during prediction. This makes it particularly effective in handling small sample and non-stationary data.

In summary, GRNN is a robust neural network model, especially suitable for non-linear mapping and processing small sample data. It has found wide application in various fields, including pattern recognition, fault prediction, and time series forecasting, among others.

In the training samples, the number of neurons in the input layer is equal to the dimension of the input vector, and each neuron serves as a simple transmission unit, directly passing the input variables to the pattern layer. The number of neurons in the pattern layer is equal to the number of learning samples, and each neuron corresponds to a different learning sample. In the pattern layer, each neuron corresponds to a transfer function, as depicted in Equation (1) [33].

$$P_i = \exp \left[ -\frac{(X - X_i)^T (X - X_i)}{2\sigma^2} \right], \quad i = 1, 2, \dots, n \tag{1}$$

where  $P_i$  represents the output of each neuron in the pattern layer,  $X$  denotes the input variables of the network,  $X_i$  corresponds to the learning sample associated with each neuron in the pattern layer, and  $\sigma$  is the smoothing factor,  $n$  is the number of samples.

In the GRNN's summation layer, there are two types of neurons that perform summation operations.

The first type is to perform arithmetic summation on the outputs of all neurons in the pattern layer, where each neuron in the pattern layer is connected to the summation neuron with a connection weight of 1. The corresponding transfer function is shown in Equation (2) [33].

$$S_D = \sum_{i=1}^n P_i \quad (2)$$

where  $P_i$  represents the output of each neuron in the pattern layer,  $n$  is the number of samples.

The second type is to perform a weighted summation of the outputs of all neurons in the pattern layer, the corresponding transfer function is shown in Equation (3) [33].

$$S_N = \sum_{i=1}^n y_i P_i \quad (3)$$

where the output value  $y_i$  of each neuron in the pattern layer serves as its connection weight,  $P_i$  represents the output of each neuron in the pattern layer,  $n$  is the number of samples.

Finally, the output  $Y$  can be obtained using Equation (4) [33].

$$Y = \frac{S_N}{S_D} \quad (4)$$

During regression analysis, GRNN predicts the output using Equation (5) [33].

$$y = E[y/X] = \frac{\int_{-\infty}^{\infty} y f(X, y) dy}{\int_{-\infty}^{\infty} f(X, y) dy} \quad (5)$$

where  $X$  represents the independent variable,  $y$  is the dependent variable, and  $f(X, y)$  is the joint probability density function of the two variables  $X$  and  $y$ . The Parzen-Window non-parametric estimation method can be used to estimate  $f(X, y)$  using random variables  $X_i$  and  $y_i$ , the method is calculated as shown in Equation (6) [33].

$$f(X, y) = \frac{1}{(2\pi)^{(P+1)/2} \sigma^{(P+1)}} \times \frac{1}{n} \sum_{i=1}^n \exp\left[-\frac{(X-X_i)^T(X-X_i)}{2\sigma^2}\right] \exp\left[-\frac{(y-y_i)^2}{2\sigma^2}\right] \quad (6)$$

where  $n$  is the number of samples,  $P$  is the dimension of the random variable  $X$ ,  $\sigma$  is the smoothing factor,  $X$  denotes the input variables of the network, and  $X_i$  corresponds to the learning sample associated with each neuron in the pattern layer.

The prediction output is given by Equation (7) [33].

$$\hat{y}(X) = \frac{\sum_{i=1}^n \exp\left[-\frac{D_i^2}{2\sigma^2}\right]}{\sum_{i=1}^n \exp\left[-\frac{D_i^2}{2\sigma^2}\right]} \quad (7)$$

where  $D_i^2 = (X - X_i)^T (X - X_i)$

As can be observed, the prediction accuracy of GRNN is influenced by the value of  $\sigma$  and it requires optimization. When  $\sigma$  is sufficiently large, the predicted results approach the mean value of  $y$ , leading to poor fitting capability to the training data. On the other hand, when  $\sigma$  is very small, the predicted results closely match the experimental  $y$  values in the training samples, but the prediction accuracy for external test data is compromised. However, when  $\sigma$  is appropriately chosen, it considers the experimental  $y$  values of all training samples and assigns larger weights to variables  $y$  corresponding to points near  $X$ , resulting in excellent prediction performance.

The metaheuristic algorithm is an excellent method for determining the value of  $\sigma$  [34]. Therefore, we develop an enhanced metaheuristic algorithm. The following is a detailed introduction.



### 3.2. Proposed Enhanced Algorithm

To train the network more efficiently and avoid getting stuck in local optima, we designed the ESCSO. We selected the SCSO algorithm as the foundation for enhancing the network for several reasons. Firstly, the SCSO algorithm has demonstrated impressive performance across diverse fields, particularly in intrusion detection systems and medical diagnosis, showcasing its adaptability and versatility [35–37]. Secondly, the algorithm's robust global search capability, inspired by biological symbiosis mechanisms, enables swift identification of global optimal solutions in complex problem spaces, mitigating the risk of getting stuck in local optima. Moreover, through diversity exploration and adaptive parameter adjustments, the algorithm effectively escapes local extremities, comprehensively exploring potential solution spaces and enhancing the likelihood of discovering global optimal solutions. Finally, during comparison with other meta-heuristic algorithms, such as particle swarm optimization algorithm, grey wolf optimization algorithm, whale optimization algorithm, etc., SCSO demonstrates better performance in several test cases and real engineering problems. Given these considerations, we are confident that utilizing the SCSO algorithm as the foundation for network improvement will significantly enhance network training efficiency, accelerate convergence towards global optimal solutions, and yield faster, more accurate optimization outcomes for fault warning applications. Additionally, we have introduced a range of novel optimization operators to further enhance its efficiency and achieve superior fault warning capabilities.

#### 3.2.1. Population Initialization

To enhance the diversity of the initial population, we use the Tent chaotic mapping method, which has good traversal uniformity, to initialize the population for ESCSO. Tent mapping is an excellent strategy to enhance the solution quality of metaheuristic algorithms and has been widely applied in other algorithms. For instance, Li et al. applied it to the whale optimization algorithm [38], Chen et al. used it to improve the GA [39], and Gao et al. combined it with the Harris Hawks optimization algorithm [40]. Furthermore, it has been extensively employed to enhance other notable algorithms such as particle swarm optimization [41], equilibrium optimizer [42], and artificial bee colony algorithm [43]. As the tent mapping may have small cycles and unstable periodic points, we introduce a random variable  $rand(0, 1) \times 1/N$  into the Tent mapping function to improve it. The updated expression is described in Equation (8).

$$y_{new(i)} = \begin{cases} 2y_{old(i)} + rand(0, 1) \times \frac{1}{N} & 0 \leq y_{old(i)} < \frac{1}{2} \\ 2(1 - y_{old(i)}) + rand(0, 1) \times \frac{1}{N} & \frac{1}{2} < y_{old(i)} \leq 1 \end{cases} \quad (8)$$

where  $y_{old(i)}$  is the initial solution after random generation,  $rand(0, 1)$  represents a random number between 0 and 1, and  $N$  is the number of particles in the Tent sequence,  $y_{new(i)}$  is the solution after mapping?

Next, we derive the individuals within the population by applying the process of inverse mapping to the uniformly distributed chaotic sequence generated by the Tent mapping, as described in Equation (9).

$$x_i = y_{new(i)}(ub - lb) + lb \quad (9)$$

where  $y_i$  is the chaotic sequence generated by Equation (8),  $x_i$  is the mapped ESCSO individual, and  $ub$  and  $lb$  represent the upper and lower bounds of the search space, respectively.

Generally, the value of  $\sigma$  is set to be between (0, 1), so we also choose its value range to be (0, 1), and we first randomly generate  $y_{old(i)}$  in the (0, 1) search space, and then chaotically map it according to Equation (8) to generate a new mapped particle  $y_{new(i)}$ , and then. Finally, we generate the final initial solution  $x_i$  according to Equation (9). For all the subsequent calculations, the value of  $\sigma$  takes the value between (0, 1).

### 3.2.2. Sand Cat Searching for Prey (Exploration Phase)

In the ESCSO algorithm, the sand cats update their positions based on the current best position of the population, their own positions, and the sensitivity range. This allows them to explore other potentially better prey positions [35]. To ensure that sand cats find new local optima within the search space, new positions are generated between the current position and the prey position. Randomness is introduced to balance the convergence effectiveness and complexity of the algorithm [35]. This phase is organized in a mathematical model as described in Equation (10) [35].

$$x_{new(i)} = r_2 \cdot \left( x_{best(t)} + rand(0,1) \cdot x_{old(i)} \right) \quad (10)$$

where  $x_{new(i)}$  denotes the new individual,  $r_2$  is the different sensitivity ranges for each sand cat, which is computed as described in Section 3.2.4,  $rand(0,1)$  denotes a random number between  $[0, 1]$ ,  $x_{old(i)}$  denotes the individual that needs to be updated, and  $x_{best(t)}$  denotes the best individual in the current iteration.

### 3.2.3. Sand Cat Attacking Prey (Exploitation Phase)

In the exploitation phase of the ESCSO algorithm, the sand cats represent the distance between the best position and the current position [35]. This allows the sand cats to explore other potentially better prey positions. The position update formula for this phase is represented as Equation (11) [35].

$$\begin{cases} P = |rand(0,1) \cdot x_{best(t)} - x_{old(i)}| \\ x_{new(i)} = x_{best(t)} - r_2 \cdot P \cdot \cos(\theta) \end{cases} \quad (11)$$

where  $P$  denotes update factor,  $rand(0,1)$  denotes a random number between  $[0, 1]$ ,  $x_{old(i)}$  denotes the individual that needs to be updated, and  $x_{best(t)}$  denotes the best individual in the current iteration,  $x_{new(i)}$  denotes the new individual. Suppose that the sensitivity range of each sand cat forms a circle so that the direction of movement can be defined by a random Angle of  $\theta$  on the circle.  $\theta$  is a random Angle between 0 and 360 degrees, i.e., the value of  $\cos(\theta)$  will be between  $-1$  and  $1$ . This allows each individual in the population to move along different circular directions within the search space. The ESCSO selects a random angle for each sand cat, enabling them to approach the prey position.

By utilizing the position update formula and selecting random angles  $\theta$ , the ESCSO enables the sand cats to explore new local optima that lie between their current positions and the prey position. This strategy increases the algorithm's exploratory capabilities, avoids getting trapped in local optima, and enhances the chances of finding global optima. Moreover, the introduction of randomness promotes diversity in the algorithm and prevents premature convergence.

### 3.2.4. Control of the Exploration and Exploitation Phase

In the ESCSO algorithm, the exploration and exploitation phases are controlled using the parameters  $r_G$  and  $R$ . These parameters allow the sand cats to transition between the two phases. The parameter  $r$  is inspired by the sand cats' low-frequency detection ability and is used to adjust the sensitivity range. Specifically, the calculation of  $r_G$  is as shown in Equation (12) [35].

$$r_G = s_M - \left( \frac{2 \times s_M \times t}{2 \times T} \right) \quad (12)$$

where  $t$  represents the current iteration count and  $T$  is the maximum iteration count. The value of  $r_G$  decreases linearly from 2 to 0 as the iterations progress, approximating the sensitivity of sand cats during hunting,  $s_M$  is inspired by sand cat hearing characteristics in SCSO assuming that it has a value of 2.  $r_G$  is the control parameter and is obtained according to Equation (12) [35].

In the ESCSO algorithm, the main parameter controlling the transition between the exploration and exploitation phases is  $R$ .  $R$  is calculated as represented in Equation (13).

$$R = 2 \times r_G \times rand(0, 1) - r_G \quad (13)$$

where  $rand(0, 1)$  represents a random number between 0 and 1, the value of  $r_G$  decreases linearly from 2 to 0 as the iterations progress,  $r_G$  is the control parameter and is obtained according to Equation (12).

In each search step, the position update of each current search agent is based on a random position, allowing them to explore new regions in the search space and avoid getting stuck in local optima. Each sand cat has a different sensitivity range, which is determined by the Equation (14) [35].

$$r_2 = r_G \times rand(0, 1) \quad (14)$$

where  $r_2$  denotes different sensitivity ranges for each sand cat.

The ESCSO ensures exploration and exploitation through the parameters  $r_G$  and  $R$ . These two parameters allow the algorithm to switch between the two phases. As the value of  $R$  depends on  $r_G$ , its fluctuation range also decreases. When the value of  $r_G$  is uniformly distributed, the value of  $R$  is well balanced, and the execution of the two phases is adaptively adjusted according to the problem. In other words,  $R$  is a random value in the interval  $[-2r_G, 2r_G]$ , and  $r$  linearly decreases from 2 to 0 during the iteration process. When the random value of  $R$  is within the range  $[-1, 1]$ , the next position of the sand cat can be at any point between its current position and the prey position. If  $R$  is less than or equal to 1, the ESCSO algorithm performs exploitation; otherwise, it is forced to perform exploration. Equation (15) demonstrates the above mechanism [35].

$$x_{new(i)} = \begin{cases} x_{best(t)} - r_2 \cdot P \cdot \cos(\theta), & \text{if } |R| \leq 1, \text{ exploitation} \\ r_2 \cdot (x_{best(t)} + rand(0, 1) \cdot x_{old(i)}), & \text{else, exploration} \end{cases} \quad (15)$$

In the ESCSO algorithm, if the value of  $R$  is less than or equal to 1 (i.e.,  $R \leq 1$ ), the algorithm enters the exploitation phase, indicating that the sand cats are guided to attack their prey. When the value of  $R$  is greater than 1 (i.e.,  $R > 1$ ), the algorithm enters the exploration phase, indicating that the sand cats' task is to search for new solutions in the local region.

### 3.2.5. Proposed Strategies for Improvement

In order to further enhance the performance of SCSO, apart from initialization based on chaotic mapping, we propose two improvement strategies, namely the elite inverse learning strategy and the Cauchy mutation strategy.

#### (1) Elite inverse learning strategy

Let the extremal point of the ordinary individual in the current population be an elite individual, then its inverse solution is defined as Equation (16).

$$x_{new(i)} = \delta(lb_t + ub_t) - x_{old(i)} \quad (16)$$

where  $\delta$  is a random value on the interval  $[0, 1]$ , and  $lb_t$  and  $ub_t$  are the maximum and minimum values in the current iteration, respectively, which change dynamically with the number of iterations,  $x_{new(i)}$  denotes the new individual,  $x_{old(i)}$  denotes the individual that needs to be updated.

#### (2) Cauchy mutation

The Cauchy mutation originates from the Cauchy distribution, which is a continuous probability distribution with no mathematical expectation, and its probability density

function is:  $f(x) = \frac{1}{\pi} \times \frac{1}{1+x^2}$ . The Cauchy distribution has characteristics such as a long step length, long tails at both ends, and compact distribution. Therefore, it is easy to generate random numbers from the origin, and it can produce a larger range of random numbers than the Gaussian mutation. Introducing the Cauchy operator into the target position update can take advantage of the regulating capability of the Cauchy operator, thereby enhancing the algorithm's ability to escape from local optima. The strategy is calculated as shown in Equation (17).

$$x_{new(i)} = cauchy(0,1) + (x_{best(t)} - x_{old(i)}) \quad (17)$$

where  $x_{new(i)}$  denotes the position of particle  $i$  in a new iteration,  $cauchy(0,1)$  is a random variable obeying a Cauchy distribution with a position parameter of 0 and a scale parameter of 1. Random numbers generated by the Cauchy distribution are used to increase the stochasticity of the optimization process and are sometimes used to improve the global search.  $x_{best(t)}$  denotes the optimal position found in  $t$  iterations.  $x_{old(i)}$  denotes the current solution's current position.

In order to improve the optimization performance of ESCSO, ESCSO adopts a dynamic selection strategy to update the target position. It alternately selects the elite inverse learning strategy and the Cauchy mutation strategy to update the target position under a certain probability  $pr$ .  $pr$  is a parameter that adapts with iteration, and different strategies are selected to update the target position using the probability  $pr$ . Its calculation method is shown in Equation (18).

$$pr = \alpha * exp^{-t/T} \quad (18)$$

where  $\alpha$  is a given parameter,  $t$  is the current number of iterations, and  $T$  is the maximum number of iterations.

### 3.2.6. ESCSO Core Framework

According to the above calculation method, the core framework of ESCSO is as following Algorithm 1:

---

#### Algorithm 1: ESCSO pseudo-code

---

```

Input:  $Npop, t, \alpha$ 
For  $i = 1: Npop$ 
    Generate an individual  $x_i$  using chaotic mapping
    Calculate the fitness value of individual  $i$ 
End for
 $t = 1$ 
While  $t \leq T$ 
    For  $i = 1: Npop$ 
        Randomly select an angle  $\theta$  between 0 and 360 degrees for each individual
        If  $R \leq 1$ 
            Update the current individual using Equation (10)
        Else
            Update the current individual using Equation (11)
        End if
        Update parameters  $r_2, r_G,$  and  $R$ 
    End for
    Calculate  $pr$ 
    For  $i = 1: Npop$ 
        If  $pr < rand$ 
            Update the individual using Equation (17)
        Else
            Update the individual using Equation (18)
        End if
    End for
    Update  $pr$ 
     $t = t + 1$ 
End while

```

---

### 3.3. ESCSO-GRNN Flowchart

Using the ESCSO method, the optimal smoothing factor  $\sigma$  of the GRNN neural network is first obtained. Then, this parameter is used to optimize the GRNN neural network, and finally, the ESCSO-GRNN neural network fault warning model is trained. The root mean square error (RMSE) of the true and predicted values is used as the fitness function, and the RMSE is calculated as shown in Equation (19).

$$f(x) = \sqrt{\frac{\sum_{i=1}^M (\hat{y}_i - y_i)^2}{M}} \quad (19)$$

where  $f(x)$  represents the fitness value, i.e., RMSE,  $M$  represents the number of indicators,  $y_i$  represents the true value, and  $\hat{y}_i$  represents the predicted value.

Finally, the specific modelling steps are shown in Figure 3.

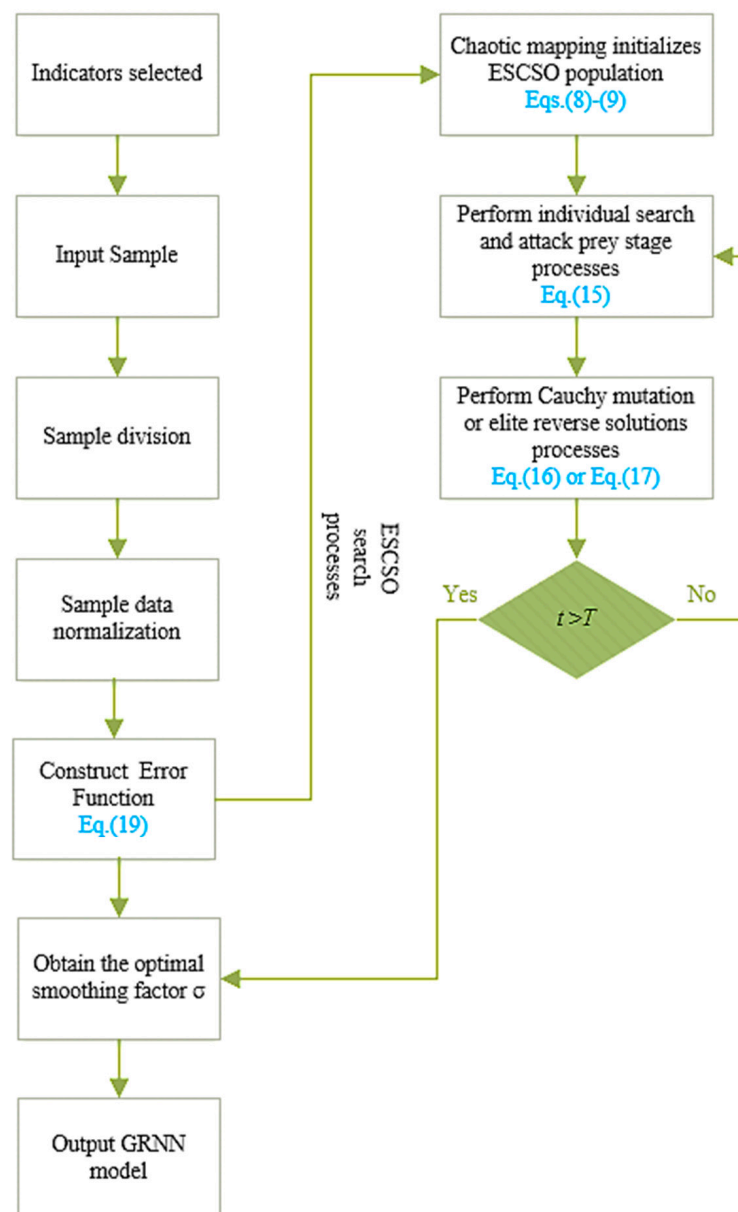


Figure 3. ESCSO-GRNN flowchart.



#### 4. Experimental Results

In this section, we validate the proposed method in a real industrial system; the results demonstrate that the method exhibits significant advantages in fault warning tasks, providing more accurate fault prediction and better generalization ability. We first introduce the data source and experimental process (Section 4.1), then perform algorithm parameters calibration (Section 4.2), followed by network training and fault warning performance testing (Section 4.3). All experiments are conducted in the environment of Table 2.

**Table 2.** Experimental environment display. <sup>1</sup> MATLAB Online—MATLAB & Simulink ([mathworks.cn](https://www.mathworks.cn)).

| Software <sup>1</sup> | MATLAB R2018b   |
|-----------------------|-----------------|
| Operating System      | Windows         |
| CPU                   | Intel® Core™ i7 |
| CPU Clock Speed       | 2.5 GHz         |
| Architecture          | 64-bit          |
| Memory                | 8 GB            |

##### 4.1. Data Source and Experimental Process

To validate the effectiveness of the proposed ESCSO-GRNN model, we conduct experiments in a certain hydroelectric power plant, focusing on the fault warning of a specific generator. We collect data for three major fault types of the generator, comprising 60 instances for each fault type, along with 1180 instances of normal operating data for the main measurement points when no faults occur. Using this dataset of 1180 instances, we train the ESCSO-GRNN model. In the validation phase, we utilize the fault data to evaluate the performance of our trained ESCSO-GRNN model in detecting and predicting faults. By comparing the model's predictions with the actual fault conditions, we can assess its accuracy and reliability. This experimental design allows us to verify the effectiveness of the ESCSO-GRNN model for fault warning and its ability to accurately predict and discriminate different fault types in the generator. It should be noted that our data is sourced from real-time databases in power plants. Temperature sensors typically convert temperature changes into electrical signals through variations in thermocouple resistance, enabling temperature data acquisition. In this context, we adopted the most common voltage divider circuit. Once the power supply and voltage divider resistor (R1) are determined, the relationship between output voltage and temperature can be established. By selecting appropriate voltage divider resistors and referring to the R-T table, voltage values corresponding to each temperature are computed. Vibration sensors primarily consist of acceleration sensors, velocity sensors, and displacement sensors used respectively for measuring vibration acceleration, vibration velocity, and vibration displacement. The current transformer method is a non-contact measurement technique that employs current transformers to convert current signals into low-level signals for subsequent measurement. This method does not require direct contact with the motor, resulting in minimal measurement error. The specific steps are as follows: Install the current transformer on the motor's circuit and connect the low-level signal output from the transformer to an ammeter or data collector. Then, the current value can be obtained and analyzed through the ammeter or data collector for further processing.

We take three fault types of a generator stator as an example, and the main measuring points of these three fault types are shown in Table 3.

**Table 3.** Types of generator fault.

| Fault Types              | Main Measurement Points  | Fault Cause  |
|--------------------------|--|--|
| Loose stator core        | Stator core<br>horizontal vibration<br>Vertical vibration of stator core<br>Stator core temperature                                | Material quality is not qualified,<br>size error,<br>silicon steel sheet aging,<br>improper design,<br>production process problems,<br>poor installation quality |
| Loose stator tooth plate | Stator core vertical vibration<br>Stator core<br>horizontal vibration<br>Stator tooth plate temperature<br>Stator core temperature | Long-term poor<br>environmental operation<br>High-temperature thermal<br>fatigue, frequent start and stop,<br>overload operation                                 |
| Stator overload          | Stator current<br>Stator winding temperature<br>Stator core temperature<br>Stator hot air temperature                              | Excessive resistance,<br>poor wiring of stator winding,<br>aging of insulation   |

#### 4.2. ESCSO-GRNN Parameters Calibration

ESCSO-GRNN is a complex neural network model, and to ensure its optimal performance in fault warning tasks, we need to calibrate its parameters. Parameters calibration is a crucial step to fine-tune the model [44–46], making it better adapt to training data and possess stronger generalization capability [47]. As described in Section 3, the ESCSO-GRNN model has three parameters:  $N_{pop}$ ,  $T$ , and  $\alpha$ . Based on preliminary experiments and literature analysis [35–39], we set four levels for each parameter, as shown in Table 4.

**Table 4.** Parameters reference level.

| Parameters | Level 1 | Level 2 | Level 3 | Level 4 |
|------------|---------|---------|---------|---------|
| $N_{pop}$  | 20      | 30      | 40      | 50      |
| $T$        | 50      | 100     | 150     | 200     |
| $\alpha$   | 0.4     | 0.45    | 0.5     | 0.55    |

Performing a comprehensive evaluation of all parameter combinations requires  $3^4 = 81$  experiments, which can be a significant computational burden. To efficiently perform parameters calibration, we decide to use the Taguchi method to form an orthogonal array, allowing us to evaluate the performance of ESCSO-GRNN parameters in relatively fewer experiments. Moreover, we use the relative percentage deviation ( $RPD$ ) as an evaluation metric to determine the performance of the ESCSO-GRNN model under different parameter combinations.

The Taguchi method is an experimental design approach that constructs orthogonal arrays, enabling a comprehensive understanding of parameters influences with fewer experiments [45]. In this case, we use the L16 orthogonal array of the Taguchi method, which means we only need 16 experiments to assess the performance of ESCSO-GRNN under different parameter combinations.

Using the relative percentage deviation ( $RPD$ ) to evaluate ESCSO-GRNN's performance is a common practice. The  $RPD$  calculation formula is shown in Equation (20), helping us measure the accuracy of the model's predictions. By calculating  $RPD$  for different parameter combinations, we can find the optimal parameter settings that optimize the ESCSO-GRNN model.

$$RPD = \frac{Alg_{Sol} - Min_{Sol}}{Min_{Sol}} \quad (20)$$

where  $Min_{Sol}$  is the minimum value from the algorithm in all the experiments and  $Alg_{Sol}$  is the value from the algorithm in each experiment.

After implementing the Taguchi method, we only need to perform 16 experiments, saving a considerable amount of computational resources compared to the full 81 experiments while still obtaining a thorough evaluation of the ESCSO-GRNN parameters calibration. This approach allows us to efficiently optimize the model, providing accurate and reliable prediction capabilities for fault warning tasks in the hydroelectric power plant generator.

Additionally, for fair and effective parameter calibration, we use the K-fold cross-validation method. This method helps us fully utilize the limited data resources, comprehensively evaluate the model's performance, and reduce the bias caused by different dataset partitions. In K-fold cross-validation, we divide the original dataset into K subsets, where  $K - 1$  subsets are used as the training data, and the remaining one subset is used as the test data [47]. This process is repeated K times, with each subset serving as the test set once while the others are training sets [47]. Finally, the results of K experiments are averaged to obtain the model's performance evaluation metric. The advantage of using K-fold cross-validation is that all data is thoroughly utilized, and each sample has an opportunity to be both training and testing data. This helps reduce the fluctuation of evaluation results due to different data splits, enhancing evaluation stability and reliability.

By carefully calibrating the ESCSO-GRNN parameters and utilizing techniques such as cross-validation, we can find the best parameter combination that allows the model to perform well on both the training and test sets, thereby enhancing the accuracy and reliability of fault warnings. This process may require multiple iterations and attempts, but the ESCSO-GRNN model, once calibrated, becomes a more powerful and effective tool, providing robust support for fault warning in hydroelectric power plant generators.

Combining the Taguchi experimental method and K-fold cross-validation, our parameter calibration process follows these specific steps:

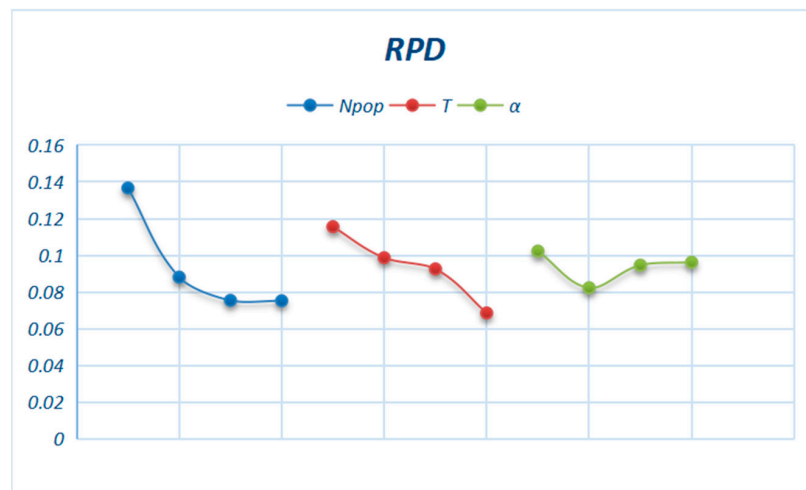
- Divide the dataset into  $K = 5$  subsets.
- For each parameter combination, conduct five calculations, using one subset as the test set and the other four subsets as the training set each time.
- Record the results for each calculation.
- Take the average of the five calculations to obtain the model performance metric for that parameter combination.
- Use the average to calculate the *RPD* for evaluating model performance.

The final results of the L16 experiments are shown in Table 5.

**Table 5.** Results of the Taguchi experiments.

| Experiment No. | <i>Npop</i> | <i>T</i> | $\alpha$ | <i>RPD</i> |
|----------------|-------------|----------|----------|------------|
| 1              | 1           | 1        | 1        | 0.15879    |
| 2              | 1           | 2        | 2        | 0.13800    |
| 3              | 1           | 3        | 3        | 0.117996   |
| 4              | 1           | 4        | 4        | 0.131631   |
| 5              | 2           | 1        | 2        | 0.11995    |
| 6              | 2           | 2        | 1        | 0.07946    |
| 7              | 2           | 3        | 4        | 0.07176    |
| 8              | 2           | 4        | 3        | 0.08137    |
| 9              | 3           | 1        | 3        | 0.09732    |
| 10             | 3           | 2        | 4        | 0.09540    |
| 11             | 3           | 3        | 1        | 0.10928    |
| 12             | 3           | 4        | 2        | 0          |
| 13             | 4           | 1        | 4        | 0.08622    |
| 14             | 4           | 2        | 3        | 0.08190    |
| 15             | 4           | 3        | 2        | 0.07140    |
| 16             | 4           | 4        | 1        | 0.06161    |

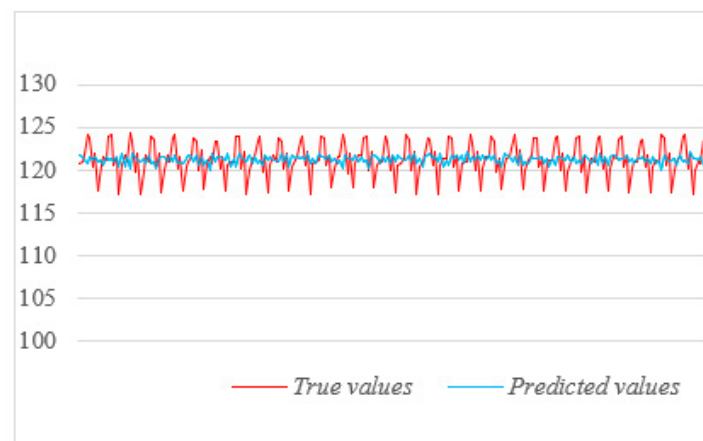
Finally, after obtaining the average values of each parameter level across different experiments, the calibration results for each parameter are shown in Figure 4.



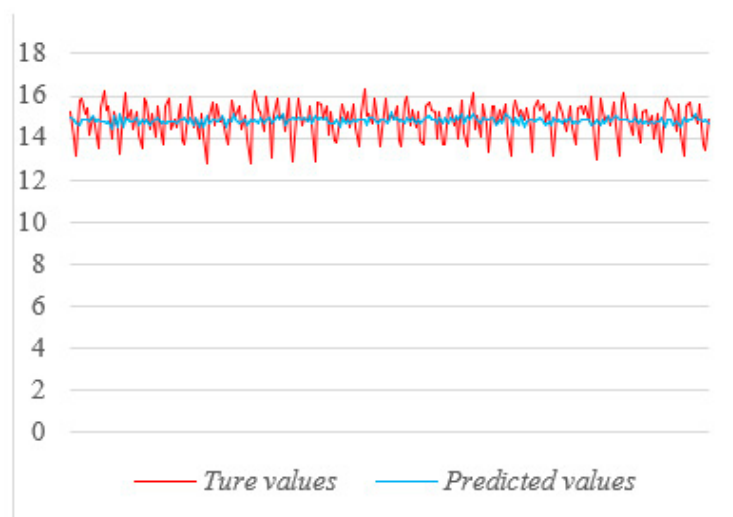
**Figure 4.** Parameters calibration results.

#### 4.3. Model Training and Fault Warning Testing

According to the results in Figure 5, we set  $N_{pop} = 50$ ,  $T = 200$ , and  $\alpha = 0.45$ . We divide all the data into a 7:3 ratio for the training and testing datasets, respectively, to conduct network training. The predicted results are shown in Figures 5–15.



**Figure 5.** Stator core horizontal vibration prediction results (Loose stator core).



**Figure 6.** Stator core vertical vibration prediction results (Loose stator core).

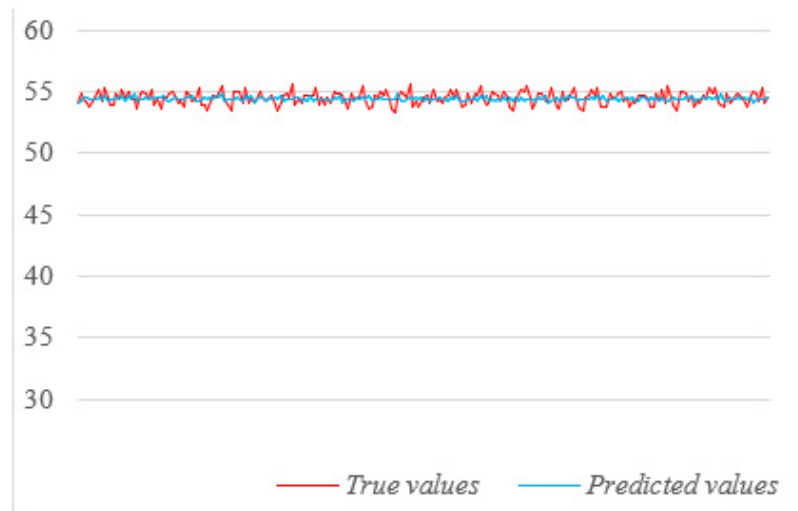


Figure 7. Stator core temperature prediction results (Loose stator core).

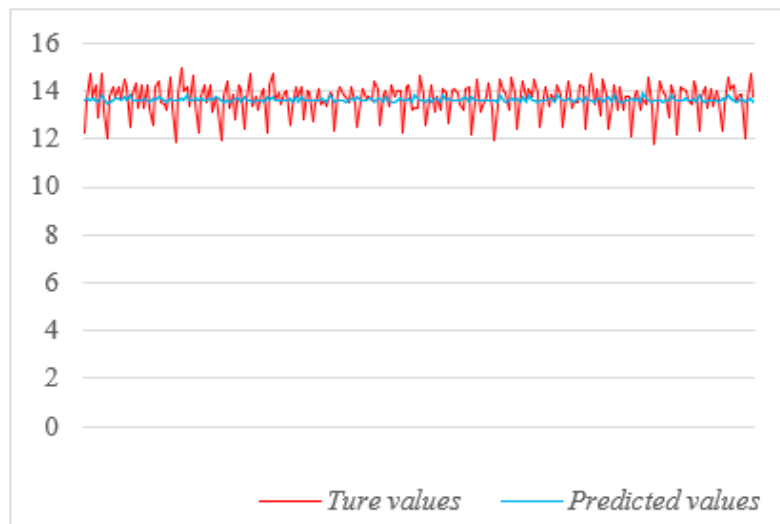


Figure 8. Stator core vertical vibration prediction results (Loose stator tooth plate).

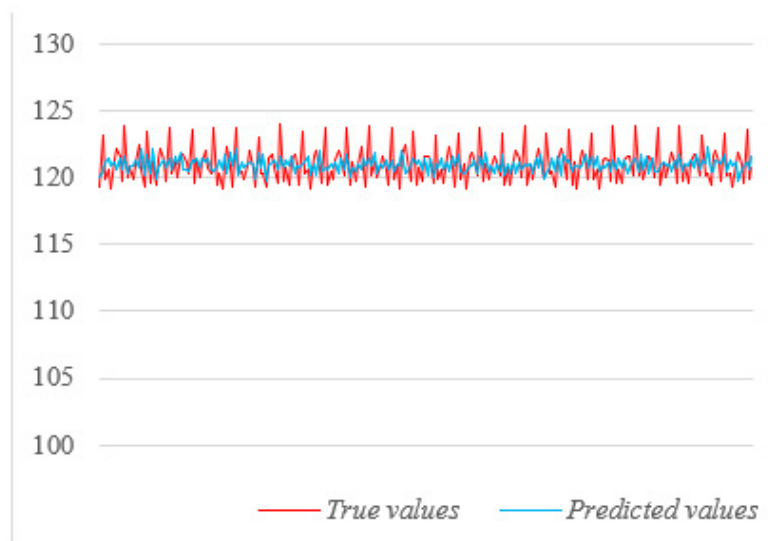
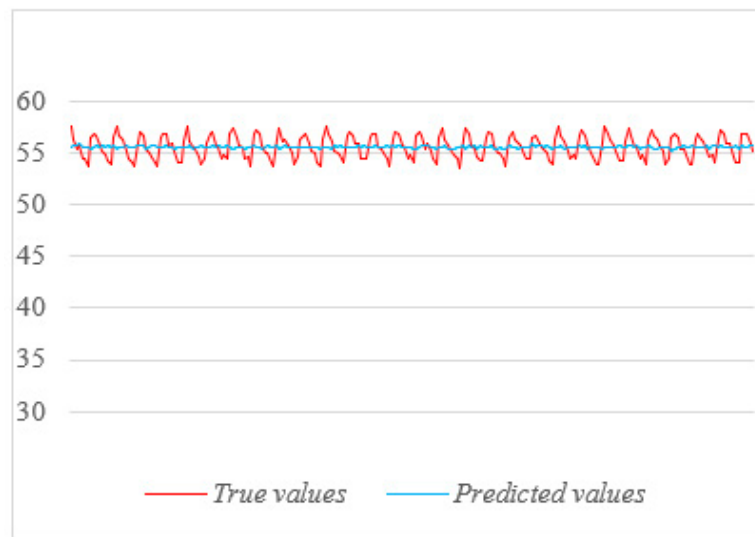
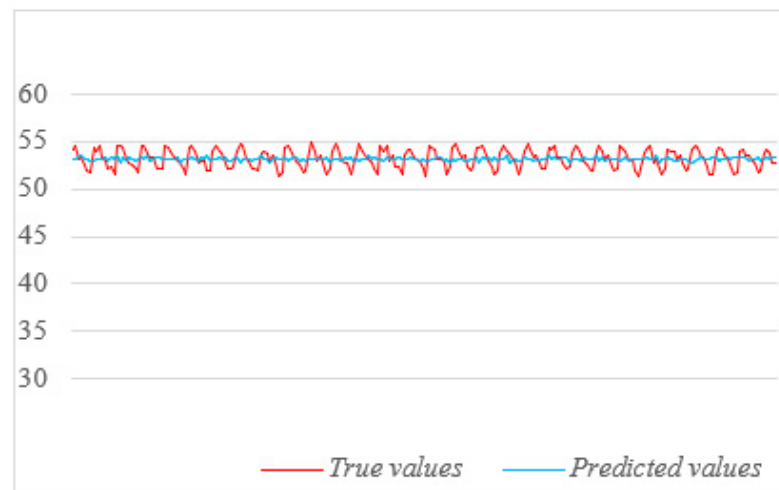


Figure 9. Stator core horizontal vibration prediction results (Loose stator tooth plate).

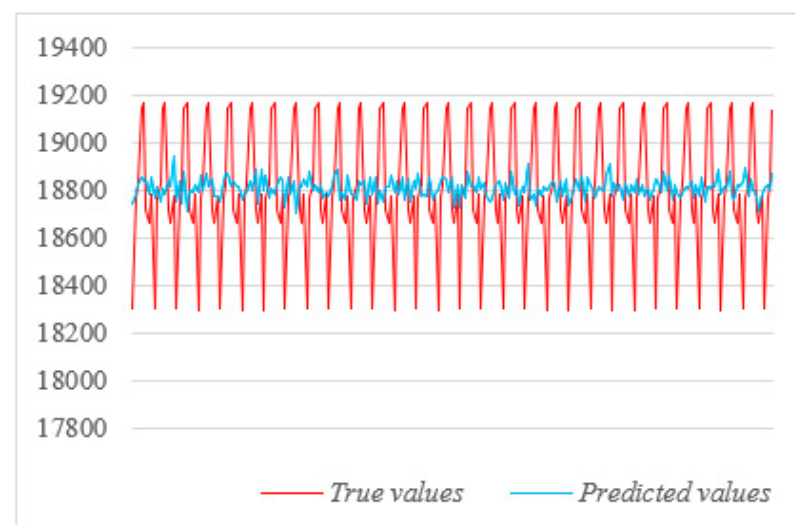




**Figure 10.** Stator temperature prediction results (Loose stator tooth plate).



**Figure 11.** Stator core temperature prediction results (Loose stator tooth plate).



**Figure 12.** Stator current prediction results (Stator overload).

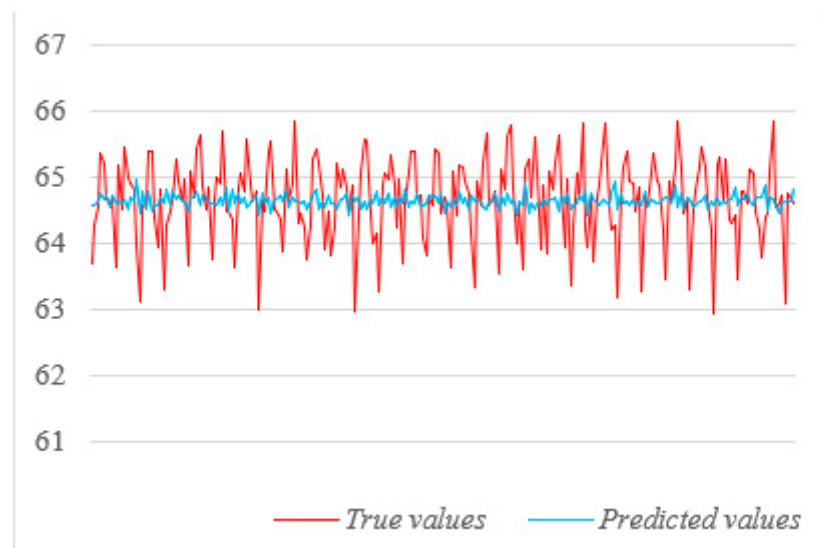


Figure 13. Stator winding temperature prediction results (Stator overload).



Figure 14. Stator core temperature prediction results (Stator overload).

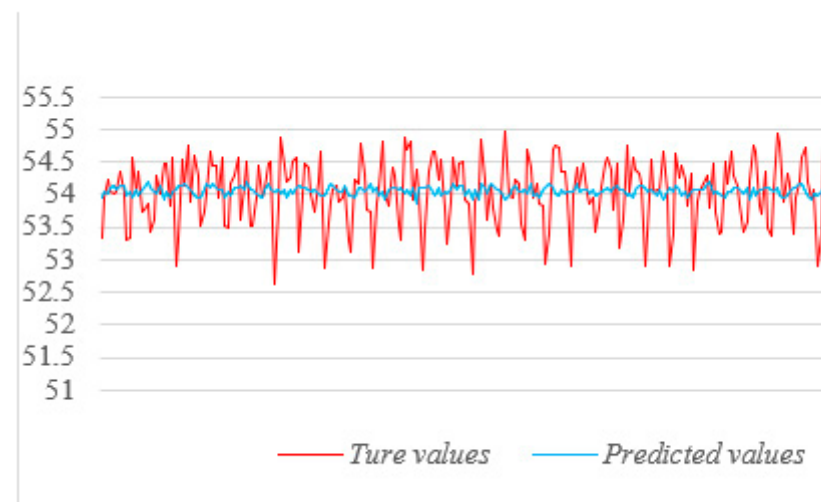


Figure 15. Stator hot air temperature prediction results (Stator overload).

Figures 5–15 show the results of the testing set for these eleven measurement points. It can be observed that our ESCSO-GRNN model achieves high accuracy and demonstrates effective fault warning capabilities.

After obtaining the predicted values, we calculate the difference between the actual values and the predicted values for the main observation points when three types of faults occur. If the difference exceeds the fault warning threshold for a specific fault type, the corresponding fault will be identified, and a warning signal will be issued.

Next, we conduct fault warning tests based on twenty sets of data collected for each of the three types of faults in the generator using the ESCSO-GRNN model. The final results are shown in Table 6.

**Table 6.** Test results of fault warning.

| Type of Faults           | Accuracy Rate |
|--------------------------|---------------|
| Loose stator core        | 53/60 (88%)   |
| Loose stator tooth plate | 56/60 (93%)   |
| Stator overload          | 55/60 (92%)   |

According to the Table 6, it is evident that our ESCSO-GRNN model performs exceptionally well in fault warning. For the “Loose stator core” and “Stator overload” fault types, the accuracy rates both exceed 85%. Furthermore, for the “Loose stator tooth plate” fault warning, the accuracy rate is an impressive 93%. These results demonstrate the high reliability and accuracy of our fault warning system in identifying different types of faults, providing effective assurance for the safe operation of the generator.

## 5. Comparison with Other Advanced Algorithms

In this section, we undertake a thorough analysis of ESCSO-GRNN. Initially, we examine its predictive efficacy (Section 5.1) and subsequently, we apply various algorithms to fault warning, comparing their respective success rates to gain comprehensive insights (Section 5.2).

### 5.1. Comparison of Predicted Efficacy

To further validate the effectiveness of our proposed ESCSO-GRNN method, we conduct performance comparisons with seven other algorithms: SSA-XGBoost [48], IEO-BP [49], MSSA-SVM [20], and GA-BP [11]. Additionally, to demonstrate the efficacy of ESCSO, we optimize GRNN using the original SCSO, social engineering optimizer [50], and GA, with their results included in the discussion. To ensure fairness, all experiments are conducted within the experimental environment outlined in Table 2. Initially, we determine the optimal parameters for all algorithms, employing a combination of K-fold cross-validation and Taguchi experimental methods for parameter calibration, as depicted in Table 7. Due to the large number of IEO-BP parameters, we use abbreviations to represent its parameters. Please refer to its literature for the meaning of abbreviations. Moreover, to mitigate the effects of algorithmic randomness during execution, we run each algorithm ten times and take the average of their performance metrics to measure the performance of these algorithms in all aspects, including RMSE, R-squared ( $R^2$ ), and computational time (CPU time).

RMSE and  $R^2$  are common metrics used to evaluate the predictive performance of models.

RMSE represents the degree of difference between predicted values and actual values. It is calculated by squaring the differences between the model’s predicted values and the observed values, taking the average, and then taking the square root of the result. A smaller RMSE indicates that the model’s predictions are closer to the actual values, indicating higher predictive accuracy.

**Table 7.** Parameters settings of the algorithms used for the comparison.

|              |   |
|--------------|---|
| IEO-BP       | $Fun1 = \text{Tanh}$ , $Fun2 = \text{ReLU}$ , $NL = 11$ , $I = 0.92$ , $T = 10$<br>$Maxit = 200$ , $Npop = 50$ , $Submit = 30$ , $T1 = 1200$ , $T0 = 100$<br>$A = 0.92$ , Number of BP network training = 1000<br>Training error = 0.02<br>Learning rate = 0.001  |
| SSA-XGBoost, | Maximum number of iterations = 200, Population size = 50, Safety value = 0.8, Percentage of discoverers is = 65%, Percentage of alerts = 30%  |
| SCSO-GRNN    | Maximum number of iterations = 200, Population size = 50  |
| GA- GRNN     | Maximum number of iterations = 200, Population size = 50, Crossover probability = 0.78, Mutation probability = 0.05   |
| SEO-GRNN     | Maximum number of iterations = 200, Population size = 50, $\alpha = 0.8$ , $\beta = \frac{\pi}{9}$  |
| GA-BP        | Maximum number of iterations = 200, Population size = 50<br>Crossover probability=0.81, Mutation probability = 0.13,<br>Input layer to implicit layer activation function = <i>Softsign</i><br>Implicit layer to output layer activation function = <i>ReLU</i><br>Number of neurons in the hidden layer = 10<br>Number of BP network training = 1000<br>Training error = 0.02<br>Learning rate = 0.001 |
| MSSA-SVM     | Maximum number of iterations = 200, Population size =50, Safety value is 0.6, Percentage of discoverers = 70%, Percentage of alerts = 20%, Number of K-fold crossings = 5   |

$R^2$  is used to measure the model's ability to explain the variability of the dependent variable. It represents the proportion of the total variance explained by the model's predicted values.  $R^2$  values range from 0 to 1, and a value closer to 1 indicates a better fit of the model and a better ability to explain the variability of the dependent variable.

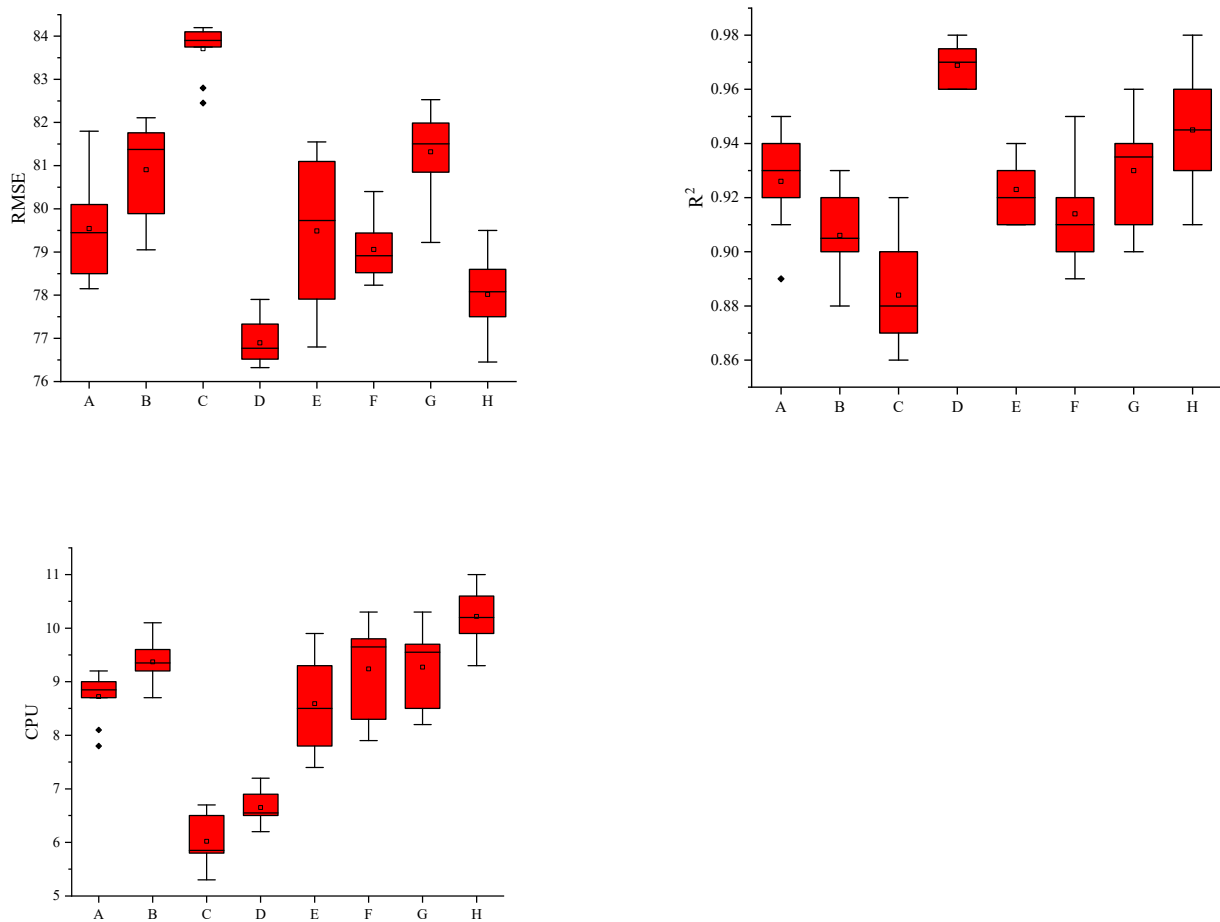
CPU time reflects the computational efficiency of each algorithm. It measures the time consumed by the algorithm during execution. A smaller CPU time indicates that the algorithm executes faster and completes the task in a shorter time.

In performance comparison, smaller values of RMSE and larger values of  $R^2$  are generally desired, as they indicate more accurate model predictions and better fit. Additionally, minimizing CPU time is preferred, as it indicates higher computational efficiency, allowing the algorithm to complete tasks in a shorter time. Therefore, in fault warning tasks, we aim for models with low RMSE, high  $R^2$ , and reduced CPU time to achieve efficient and accurate fault warning models.

The average results of the ten runs are summarized in Table 8 and visualized in Figure 16.

**Table 8.** Results of algorithms performance comparison.

| Algorithms  | RMSE  | R2   | CPU/s |
|-------------|-------|------|-------|
| IEO-BP      | 79.54 | 0.93 | 8.72  |
| SSA-XGBoost | 80.90 | 0.91 | 9.36  |
| SCSO-GRNN   | 83.72 | 0.88 | 6.02  |
| ESCSO-GRNN  | 76.89 | 0.97 | 6.65  |
| GA-GRNN     | 79.48 | 0.92 | 8.59  |
| SEO-GRNN    | 79.06 | 0.91 | 9.24  |
| GA-BP       | 81.32 | 0.93 | 9.27  |
| MSSA-SVM    | 78.02 | 0.95 | 10.20 |



**Figure 16.** Statistical results of algorithms performance comparison.

Based on the above results, our ESCSO-GRNN achieves the best values for RMSE and  $R^2$ . Although its CPU time is slightly larger than that of SCSO-GRNN, the predictive results of ESCSO-GRNN far exceed those of SCSO-GRNN. Additionally, in terms of stability, ESCSO-GRNN demonstrates the best stability across all three metrics, which strongly indicates the effectiveness of ESCSO-GRNN.

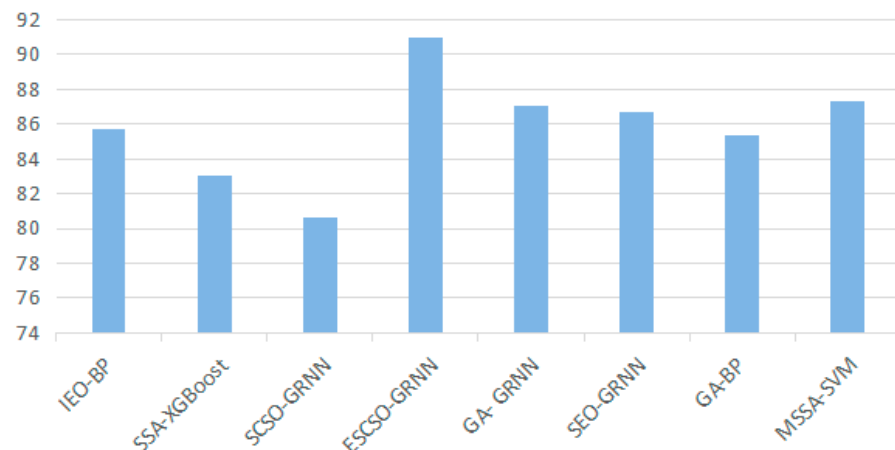
### 5.2. Comparison of Fault Warning Success Rates

Next, we compare the fault warning accuracy of the four methods, and the results can be found in Table 9. Figure 17 displays the overall fault warning accuracy for all tasks achieved by the four methods.

**Table 9.** Test results of four algorithms for fault warning.

| Type of Faults           | IEO-BP      | SSA-XGBoost | SCSO-GRNN  | ESCSO-GRNN  |
|--------------------------|-------------|-------------|------------|-------------|
| Loose stator core        | 50/60 (83%) | 51/60(85%)  | 49/60(81%) | 53/60 (88%) |
| Loose stator tooth plate | 52/60 (86%) | 50/60(83%)  | 50/60(83%) | 56/60 (93%) |
| Stator overload          | 53/60 (88%) | 49/60(81%)  | 47/60(78%) | 55/60 (92%) |
| Type of Faults           | GA-GRNN     | SEO-GRNN    | GA-BP      | MSSA-SVM    |
| Loose stator core        | 51/60 (85%) | 53/60(88%)  | 50/60(83%) | 49/60 (82%) |
| Loose stator tooth plate | 53/60 (88%) | 52/60(86%)  | 51/60(85%) | 53/60 (88%) |
| Stator overload          | 53/60 (88%) | 52/60(86%)  | 53/60(88%) | 55/60 (92%) |





**Figure 17.** The total success rate of four algorithms fault warning test.

In accordance with the data presented in Table 9, a discernible pattern emerges showcasing the accuracy performance of each individual fault prediction model across diverse fault types. Notably, the ESCSO-GRNN model consistently outshines its counterparts, attaining the highest accuracies across all three fault categories, namely 88%, 93%, and 92%. Noteworthy achievements are also evident with the MSSA-SVM model, which similarly secures the pinnacle of precision in predicting Stator overload faults. Additionally, the SEO-GRNN model aligns itself with the ESCSO-GRNN in achieving parity when addressing Loose stator core faults. These collective findings undeniably underscore the effectiveness of the proposed algorithm.

Furthermore, the graphical representation in Figure 17 serves to reinforce this point, as it vividly illustrates the substantial superiority of the ESCSO-GRNN's fault warning accuracy over the alternative three algorithms across the entire spectrum of 180 tasks. In light of these cumulative insights, it becomes evident that the ESCSO-GRNN model consistently demonstrates elevated precision in the context of this fault prediction task. Consequently, it emerges as a potent contender for efficiently executing fault warning assignments.

## 6. Conclusions and Future Work

This paper introduces a fault warning approach that combines ESCSO and GRNN to tackle challenges in complex industrial systems and growing fault risks. The method is applied to predict specific generator faults, validated with real industrial data, and demonstrates superior performance compared to state-of-the-art methods. Our approach reduces RMSE by approximately 4.68%, with a maximum of 8.88%, and increases R2 by 5.71% on average, up to 10.23%. Additionally, it excels in program execution time, showing a 26.4% average improvement, and achieves around 6.24% average improvement in fault warning success rate compared to other methods. This provides a new perspective and approach for fault early warning research in the industrial sector. It enables timely detection of anomalies and the implementation of preventive measures, reducing production interruptions and maintenance costs, while simultaneously enhancing production efficiency and equipment reliability.

However, some limitations should be acknowledged. The study's focus is limited to particular generators and industrial systems, potentially varying across types. And validation is restricted to specific systems and geographic regions. While significant accuracy improvement is achieved, the study mainly explores specific parameter optimization, overlooking other combinations' impact. In addition, the study relies on specific data, potentially affecting long-term applicability as system behavior evolves.

Future research can explore various avenues to more comprehensively validate and enhance the practicality and adaptability of our proposed method. Expanding the scope of applicability validation by conducting experiments on a broader range of industrial

equipment types and diverse geographical contexts will provide robust confirmation of the method's universal applicability across different conditions. Moreover, experimenting with improved parameter combinations could further elevate prediction performance. By systematically testing diverse parameter configurations and optimization strategies while accounting for potential dynamic influences, the method's adaptability to changing environments can be enhanced. Considering the costs associated with subsequent maintenance and updates is crucial for long-term viability, as new data, conditions, and requirements emerge, the continued adaptability of the method is pivotal [51,52]. Simplifying the proposed method could also be a focus of future research, taking into account factors like deployment complexity, resource requirements, and operational usability [53,54]. Other advanced metaheuristic algorithms, such as the novel diffusion memetic optimizer [55] and online learning-based evolutionary multi-objective algorithms [56], could be considered for hybridization with ESCSO. Integrating these developments, which are used for other complex problems, could further enhance its efficiency. Additionally, the incorporation of more evolutionary strategies into ESCSO can be explored to promote its performance [51,57,58].

**Author Contributions:** Conceptualization, Y.P. and Y.C.; Data curation, Y.X.; Formal analysis, A.-M.G. and Y.G.; Supervision, A.-M.G. and Y.G.; Validation, Y.P.; Visualization, Y.T.; Writing—original draft, Y.P.; Writing—review and editing, Y.T. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** Not applicable.

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

## References

1. Tian, G.; Zhang, L.; Fathollahi-Fard, A.M.; Kang, Q.; Li, Z.; Wong, K.Y. Addressing a collaborative maintenance planning using multiple operators by a multi-objective Metaheuristic algorithm. *IEEE Trans. Autom. Sci. Eng.* **2023**; *in press*. [[CrossRef](#)]
2. Tao, L.; Siqi, Q.; Zhaochao, M.; Gaofeng, X. Early fault warning of wind turbine based on BRNN and large sliding window. *J. Intell. Fuzzy Syst.* **2020**, *38*, 3389–3401. [[CrossRef](#)]
3. Jieyang, P.; Kimmig, A.; Dongkun, W.; Niu, Z.; Zhi, F.; Jiahai, W.; Liu, X.; Ovtcharova, J. A systematic review of data-driven approaches to fault diagnosis and early warning. *J. Intell. Manuf.* **2022**, 1–28. [[CrossRef](#)]
4. Bendu, H.; Deepak, B.B.V.L.; Murugan, S. Application of GRNN for the prediction of performance and exhaust emissions in HCCI engine using ethanol. *Energy Convers. Manag.* **2016**, *122*, 165–173. [[CrossRef](#)]
5. Polat, Ö.; Yıldırım, T. Genetic optimization of GRNN for pattern recognition without feature extraction. *Expert Syst. Appl.* **2008**, *34*, 2444–2448. [[CrossRef](#)]
6. Cheng, J.; Xiong, Y. The quality evaluation of classroom teaching based on FOA-GRNN. *Procedia Comput. Sci.* **2017**, *107*, 355–360. [[CrossRef](#)]
7. Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **1997**, *1*, 67–82. [[CrossRef](#)]
8. Wu, H.; Fu, W.; Ren, X.; Wang, H.; Wang, E. A Three-Step Framework for Multimodal Industrial Process Monitoring Based on DLAN, TSQTA, and FSN. *Processes* **2023**, *11*, 318. [[CrossRef](#)]
9. Luo, Z.; Liu, C.; Liu, S. A novel fault prediction method of wind turbine gearbox based on pair-copula construction and BP neural network. *IEEE Access* **2020**, *8*, 91924–91939. [[CrossRef](#)]
10. Chen, H.; Li, S.; Li, M. Multi-Channel High-Dimensional Data Analysis with PARAFAC-GA-BP for Nonstationary Mechanical Fault Diagnosis. *Int. J. Turbomach. Propuls. Power* **2022**, *7*, 19. [[CrossRef](#)]
11. Jiang, H.; Yu, Z.; Wang, Y.; Zhang, B.; Song, J.; Wei, J. The state prediction method of the silk dryer based on the GA-BP model. *Sci. Rep.* **2022**, *12*, 14615. [[CrossRef](#)] [[PubMed](#)]
12. Zhao, H.; Liu, H.; Hu, W.; Yan, X. Anomaly detection and fault analysis of wind turbine components based on deep learning network. *Renew. Energy* **2018**, *127*, 825–834. [[CrossRef](#)]
13. Chen, S.; Ma, Y.; Ma, L. Fault early warning of pitch system of wind turbine based on GA-BP neural network model. *E3S Web Conf.* **2020**, *194*, 03005. [[CrossRef](#)]
14. Zhang, L.; Gao, T.; Cai, G.; Hai, K.L. Research on electric vehicle charging safety warning model based on back propagation neural network optimized by improved gray wolf algorithm. *J. Energy Storage* **2022**, *49*, 104092. [[CrossRef](#)]
15. Lin, J.; Zhao, Y.; Cui, B.; Li, Z. Fault Diagnosis of Active Phase Change Control Device based on SGSSA-BP Neural Network. In Proceedings of the 2022 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 24–26 June 2022; pp. 348–353.

16. Gao, D.; Wang, Y.; Zheng, X.; Yang, Q. A fault warning method for electric vehicle charging process based on adaptive deep belief network. *World Electr. Veh. J.* **2021**, *12*, 265. [[CrossRef](#)]
17. Zhou, Y.; Kumar, A.; Parkash, C.; Vashishtha, G.; Tang, H.; Xiang, J. A novel entropy-based sparsity measure for prognosis of bearing defects and development of a sparsogram to select sensitive filtering band of an axial piston pump. *Measurement* **2022**, *203*, 111997. [[CrossRef](#)]
18. Wang, H.; Chen, J.; Zhu, X.; Song, L.; Dong, F. Early warning of reciprocating compressor valve fault based on deep learning network and multi-source information fusion. *Trans. Inst. Meas. Control* **2023**, *45*, 777–789. [[CrossRef](#)]
19. Chu, W.L.; Lin, C.J.; Kao, K.C. Fault diagnosis of a rotor and ball-bearing system using DWT integrated with SVM, GRNN, and visual dot patterns. *Sensors* **2019**, *19*, 4806. [[CrossRef](#)]
20. Ji, C.; Wang, L.; Wang, X.; Li, X.; Cao, W. Design of cable tunnel fault early warning system based on MSSA-SVM. *J. Eng. Des.* **2023**, *30*, 109–116.
21. Yang, Y.; Zhang, S.; Su, K.; Fang, R. Early warning of stator winding overheating fault of water-cooled turbogenerator based on SAE-LSTM and sliding window method. *Energy Rep.* **2023**, *9*, 199–207. [[CrossRef](#)]
22. Peng, J.; Kimmig, A.; Niu, Z.; Wang, J.; Liu, X.; Wang, D.; Ovtcharova, J. Wind turbine failure prediction and health assessment based on adaptive maximum mean discrepancy. *Int. J. Electr. Power Energy Syst.* **2022**, *134*, 107391. [[CrossRef](#)]
23. Kirbaş, I.; Kerem, A. A new vibration-based hybrid anomaly detection model for preventing high-power generator failures in power plants. *Energy Sources Part A Recovery Util. Environ. Eff.* **2021**, *43*, 3184–3202. [[CrossRef](#)]
24. Qiao, L.; Zhang, Y.; Wang, Q. Fault detection in wind turbine generators using a meta-learning-based convolutional neural network. *Mech. Syst. Signal Process.* **2023**, *200*, 110528. [[CrossRef](#)]
25. Li, J.; Liu, J.; Chen, Y. A fault warning for inter-turn short circuit of excitation winding of synchronous generator based on GRU-CNN. *Glob. Energy Interconnect.* **2022**, *5*, 236–248. [[CrossRef](#)]
26. Niu, S.-Y.; Liu, B.-W.; Zhang, X.Y. Research on Fault Warning of doubly Fed Wind Power Generator based on LS-SVM. In Proceedings of the 3rd Annual International Conference on Electronics, Electrical Engineering and Information Science (EEEIS 2017), Guangzhou, China, 8–10 September 2017; pp. 158–163.
27. Li, M.; Zhang, G.; Zhao, S. Research on the Pit Overall Stability Intelligent Forecasting and Early Warning Method Based on GRNN. *Recent Pat. Comput. Sci.* **2016**, *9*, 68–73. [[CrossRef](#)]
28. Kaminski, M.; Kowalski, C.T.; Orłowska-Kowalska, T. General regression neural networks as rotor fault detectors of the induction motor. In Proceedings of the 2010 IEEE International Conference on Industrial Technology, Vina del Mar, Chile, 14–17 March 2010; pp. 1239–1244.
29. Chen, B.; Xu, W.; Huang, Y.; Cao, G.; Guan, S.; Yue, J. A predictive model for coal mill safety assessment based on PCA-GRNN. *Clean Coal Technol.* **2022**, *28*, 206–214.
30. Liu, X.; Dong, J.; Tu, G. Research on Fan Operation Evaluation and Error State Judgment Relying on Improved Neural Network and Intelligent Computing. *J. Phys. Conf. Ser.* **2021**, *2083*, 042005. [[CrossRef](#)]
31. Qi, Y.; Jing, T.; Ren, C.; Gao, X. Three-Stage Wind Turbine Assessment Method: Condition Monitoring, Failure Prediction, And Health Assessment. *Res. Sq.* **2021**. [[CrossRef](#)]
32. Tongmei, J.I.N.G.; Yongsheng, Q.; Liqiang, L. Condition monitoring and health assessment of wind turbine gearbox based on KECA-GRNN. *Acta Energetica Solaris Sin.* **2021**, *42*, 400–408.
33. Specht, D.F. A general regression neural network. *IEEE Trans. Neural Netw.* **1991**, *2*, 568–576. [[CrossRef](#)]
34. Tian, G.; Zhang, C.; Fathollahi-Fard, A.M.; Li, Z.; Zhang, C.; Jiang, Z. An enhanced social engineering optimizer for solving an energy-efficient disassembly line balancing problem based on bucket brigades and cloud theory. *IEEE Trans. Ind. Inform.* **2022**, *19*, 7148–7159. [[CrossRef](#)]
35. Seyyedabbasi, A.; Kiani, F. Sand Cat swarm optimization: A nature-inspired algorithm to solve global optimization problems. *Eng. Comput.* **2023**, *39*, 2627–2651. [[CrossRef](#)]
36. Jovanovic, D.; Marjanovic, M.; Antonijevic, M.; Zivkovic, M.; Budimirovic, N.; Bacanin, N. Feature selection by improved sand cat swarm optimizer for intrusion detection. In Proceedings of the 2022 International Conference on Artificial Intelligence in Everything (AIE), Lefkosa, Cyprus, 2–4 August 2022; pp. 685–690.
37. Qtaish, A.; Albashish, D.; Braik, M.; Alshammari, M.T.; Alreshidi, A.; Alreshidi, E.J. Memory-based Sand Cat Swarm Optimization for Feature Selection in Medical Diagnosis. *Electronics* **2023**, *12*, 2042. [[CrossRef](#)]
38. Li, Y.; Han, M.; Guo, Q. Modified whale optimization algorithm based on tent chaotic mapping and its application in structural optimization. *KSCE J. Civ. Eng.* **2020**, *24*, 3703–3713. [[CrossRef](#)]
39. Chen, S.; Wang, S. An optimization method for an integrated energy system scheduling process based on NSGA-II improved by tent mapping chaotic algorithms. *Processes* **2020**, *8*, 426. [[CrossRef](#)]
40. Gao, Z.-M.; Zhao, J.; Hu, Y.-R.; Chen, H.F. The improved Harris hawk optimization algorithm with the Tent map. In Proceedings of the 2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), Xiamen, China, 18–20 October 2019; pp. 336–339.
41. Varol Altay, E.; Alatas, B. Bird swarm algorithms with chaotic mapping. *Artif. Intell. Rev.* **2020**, *53*, 1373–1414. [[CrossRef](#)]
42. Gao, Z.-M.; Zhao, J.; Li, S.-R.; Hu, R.-R. The improved equilibrium optimization algorithm with tent map. In Proceedings of the 2020 5th international conference on computer and communication systems (ICCCS), Shanghai, China, 15–18 May 2020; pp. 343–346.

43. Zheng, X.; Li, X.; Li, Y.; Liu, Y. An improved artificial bee Colony algorithm based on cat mapping and differential variation. *J. Data Inf. Manag.* **2022**, *4*, 119–135. [[CrossRef](#)]
44. Fathollahi-Fard, A.M.; Dulebenets, M.A.; Hajiaghahi-Keshteli, M.; Tavakkoli-Moghaddam, R.; Safaeian, M.; Mirzahosseini, H. Two hybrid meta-heuristic algorithms for a dual-channel closed-loop supply chain network design problem in the tire industry under uncertainty. *Adv. Eng. Inform.* **2021**, *50*, 101418. [[CrossRef](#)]
45. Fathollahi-Fard, A.M.; Tian, G.; Ke, H.; Fu, Y.; Wong, K.Y. Efficient Multi-objective Metaheuristic Algorithm for Sustainable Harvest Planning Problem. *Comput. Oper. Res.* **2023**, *158*, 106304. [[CrossRef](#)]
46. Zhang, X.; Zhou, H.; Fu, C.; Mi, M.; Zhan, C.; Pham, D.T.; Fathollahi-Fard, A.M. Application and planning of an energy-oriented stochastic disassembly line balancing problem. *Environ. Sci. Pollut. Res.* **2023**, 1–15. [[CrossRef](#)]
47. Ling, H.; Qian, C.; Kang, W.; Liang, C.; Chen, H. Combination of Support Vector Machine and K-Fold cross validation to predict compressive strength of concrete in marine environment. *Constr. Build. Mater.* **2019**, *206*, 355–363. [[CrossRef](#)]
48. Zhu, Y. Research on early warning of induced draft fan failure in thermal power plant based on SSA-XGBoost. *Qinghai Electr. Power* **2022**, *41*, 37–41,49.
49. Liu, J.; Zhan, C.; Wang, H.; Zhang, X.; Liang, X.; Zheng, S.; Meng, Z.; Zhou, G. Developing a Hybrid Algorithm Based on an Equilibrium Optimizer and an Improved Backpropagation Neural Network for Fault Warning. *Processes* **2023**, *11*, 1813. [[CrossRef](#)]
50. Fathollahi-Fard, A.M.; Hajiaghahi-Keshteli, M.; Tavakkoli-Moghaddam, R. The social engineering optimizer (SEO). *Eng. Appl. Artif. Intell.* **2018**, *72*, 267–293. [[CrossRef](#)]
51. Tian, G.; Lu, W.; Zhang, X.; Zhan, M.; Dulebenets, M.A.; Aleksandrov, A.; Fathollahi-Fard, A.M.; Ivanov, M. A survey of multi-criteria decision-making techniques for green logistics and low-carbon transportation systems. *Environ. Sci. Pollut. Res.* **2023**, *30*, 57279–57301. [[CrossRef](#)] [[PubMed](#)]
52. Tian, G.; Yuan, G.; Aleksandrov, A.; Zhang, T.; Li, Z.; Fathollahi-Fard, A.M.; Ivanov, M. Recycling of spent Lithium-ion Batteries: A comprehensive review for identification of main challenges and future research trends. *Sustain. Energy Technol. Assess.* **2022**, *53*, 102447. [[CrossRef](#)]
53. Qiao, L.; Wang, Z.; Zhu, J. Application of improved GRNN model to predict interlamellar spacing and mechanical properties of hypereutectoid steel. *Mater. Sci. Eng. A* **2020**, *792*, 139845. [[CrossRef](#)]
54. Chen, Y.; Shen, L.; Li, R.; Xu, X.; Hong, H.; Lin, H.; Chen, J. Quantification of interfacial energies associated with membrane fouling in a membrane bioreactor by using BP and GRNN artificial neural networks. *J. Colloid Interface Sci.* **2020**, *565*, 1–10. [[CrossRef](#)]
55. Zhao, H.; Zhang, C. An online-learning-based evolutionary many-objective algorithm. *Inf. Sci.* **2020**, *509*, 1–21. [[CrossRef](#)]
56. Pasha, J.; Nwodu, A.L.; Fathollahi-Fard, A.M.; Tian, G.; Li, Z.; Wang, H.; Dulebenets, M.A. Exact and metaheuristic algorithms for the vehicle routing problem with a factory-in-a-box in multi-objective settings. *Adv. Eng. Inform.* **2022**, *52*, 101623. [[CrossRef](#)]
57. Dulebenets, M.A. A Diffused Memetic Optimizer for reactive berth allocation and scheduling at marine container terminals in response to disruptions. *Swarm Evol. Comput.* **2023**, *80*, 101334. [[CrossRef](#)]
58. Singh, E.; Pillay, N. A study of ant-based pheromone spaces for generation constructive hyper-heuristics. *Swarm Evol. Comput.* **2022**, *72*, 101095. [[CrossRef](#)]

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