energies

MDPI

Article

Enhanced Distributed Parallel Firefly Algorithm Based on the Taguchi Method for Transformer Fault Diagnosis

Zhi-Jun Li [1,2], Wei-Gen Chen [1], Jie Shan [3,*], Zhi-Yong Yang [2] and Ling-Yan Cao [2]

[1] State Key Laboratory of Power Transmission Equipment and System Security and New Technology, Chongqing University, Chongqing 400044, China; lzj_nzkj@126.com (Z.-J.L.); weigench@cqu.edu.cn (W.-G.C.)
[2] Guodian Nanjing Automation Co., Ltd., Nanjing 210032, China; 13951718763@139.com (Z.-Y.Y.); lingyan-cao@sac-china.com (L.-Y.C.)
[3] School of Electric Power Engineering, Shanghai University of Electric Power, Shanghai 200090, China
* Correspondence: shanjie@mail.shiep.edu.cn

Abstract: To improve the reliability and accuracy of a transformer fault diagnosis model based on a backpropagation (BP) neural network, this study proposed an enhanced distributed parallel firefly algorithm based on the Taguchi method (EDPFA). First, a distributed parallel firefly algorithm (DPFA) was implemented and then the Taguchi method was used to enhance the original communication strategies in the DPFA. Second, to verify the performance of the EDPFA, this study compared the EDPFA with the firefly algorithm (FA) and DPFA under the test suite of Congress on Evolutionary Computation 2013 (CEC2013). Finally, the proposed EDPFA was applied to a transformer fault diagnosis model by training the initial parameters of the BP neural network. The experimental results showed that: (1) The Taguchi method effectively enhanced the performance of EDPFA. Compared with FA and DPFA, the proposed EDPFA had a faster convergence speed and better solution quality. (2) The proposed EDPFA improved the accuracy of transformer fault diagnosis based on the BP neural network (up to 11.11%).

Keywords: firefly algorithm; the Taguchi method; communication strategy; transformer fault diagnosis; BP neural network

Citation: Li, Z.-J.; Chen, W.-G.; Shan, J.; Yang, Z.-Y.; Cao, L.-Y. Enhanced Distributed Parallel Firefly Algorithm Based on the Taguchi Method for Transformer Fault Diagnosis. *Energies* **2022**, *15*, 3017. https://doi.org/10.3390/en15093017

Academic Editors: Davide Astolfi and Adolfo Dannier

Received: 14 February 2022
Accepted: 18 April 2022
Published: 20 April 2022

1. Introduction

Since swarm intelligence optimization algorithms were proposed, they have been accepted by more and more non-computer researchers due to their efficient optimization performance, especially because they do not need special information about the problems to be optimized [1]. Their application fields have rapidly expanded to scientific computing [2], workshop scheduling optimization [3], transportation configuration [4], combination problems [5], digital image processing [6], engineering optimization design [7] and other fields. They have become an indispensable part of artificial intelligence and computer science. However, compared with traditional optimization algorithms, the development history of swarm intelligence optimization algorithms is still relatively short and there are many imperfections. In particular, the foundation of mathematics has always been a hindrance in its development [8]. Therefore, there are still too many problems to be explored and solved in this field.

The Taguchi method is a robust industrial design method that is used to evaluate and implement improvements in products, processes and equipment [9]. It is an experimental design method that focuses on minimizing process variability or making products less sensitive to environmental variability [10]. GA is a famous optimization algorithm [11]. The genetic algorithm has good global search ability and can quickly search out all the solutions in the solution space, but the local search ability of the genetic algorithm is poor and the search efficiency is low in the late evolution [12]. Chou and his associates used the Taguchi method with the genetic algorithm (GA), which improved the quality of

solutions [13]. Furthermore, Tsai also applied the Taguchi method to parallel cat swarm optimization (PCSO) [14], which reduced the computational time. GA is a famous optimization algorithm [15]. In 2011, P. Subbaraj combined the self-adaptive real-coded GA with the Taguchi method, which exploited the potential offspring [16]. To sum up, it is a good idea to introduce the Taguchi method into swarm intelligence optimization algorithms. However, so far, no researcher has applied the Taguchi method to the parallel technology or to improve the performance of FA.

The firefly algorithm (FA) was proposed by professor Xin-she Yang and simulates the behavioral characteristics of fireflies [17]. A firefly indicates a solution to the objective problem, and these fireflies consider light intensity and attractiveness to constantly update their position to find potentially better solutions. The advantage of FA is that it has fewer control parameters and is easier to implement than other algorithms [18]. Studies [17,18] have shown that FA has excellent global optimization ability. However, it still has some defects in concrete engineering problems, such as slow convergence speed and low solution accuracy [19]. Therefore, many changes and improvements have been made to the original FA algorithm to produce, for example, the Gaussian firefly algorithm (GD-FF) [20], firefly algorithm with chaos (chaotic FA) [21], binary firefly algorithm (BFA) [22], parallel firefly algorithm (PFA) [23], distributed parallel firefly algorithm (DPFA) [24] and so on. Among them, DPFA was proposed by Pan in 2021. Compared with the standard FA, its solution accuracy is better and its convergence speed is faster. However, the communication strategies in DPFA are too simple, and the advantage of collaboration between groups of distributed parallel strategies is not fully utilized. Therefore, in this study, the Taguchi method is introduced into the communication strategies of DPFA, and an enhanced distributed parallel firefly algorithm (EDPFA) was proposed. The proposed EDPFA can further improve the convergence speed and solution accuracy of DPFA and FA, which was successfully used for transformer fault diagnosis based on a BP neural network.

The firefly algorithm has been used to solve many multiple optimization problems, such as the economic emission load dispatch problem [25], the mobile robot navigation [26], the optimal overcurrent relay coordination [27] and the aeration modeling on spillways [28]. However, there are few related studies on the application of the FA to transformer fault diagnosis. Therefore, this study aimed to use EDPFA to investigate transformer fault diagnosis.

Power transformers are the key equipment in a power system [29,30]. In actual production, whether the existing or latent transformer faults can be diagnosed or predicted quickly and accurately is closely relevant to the safety and stability of a power system [31]. In the field of transformers, dissolved gas analysis (DGA) is an important transformer fault diagnosis technique that establishes a mathematical relationship between a fault type and fault gas [32,33]. In recent years, many algorithms and theories have been proposed and utilized in the field of transformer fault diagnosis, including the fuzzy algorithm [34], support vector machines [35], clustering algorithm [36] and neural networks [37]. These methods have achieved some research results, but they more or less have some limitations, such as easily falling into the local optimal, slow convergence and only supporting a small amount of sample data. Therefore, this study constructed a transformer fault diagnosis model using a BP neural network. A BP network is a mathematical model that can approximate any complex nonlinear relationship to the maximum extent and automatically correct the parameters. However, the unreasonable initial weights and thresholds limit the performance of a BP neural network [7]. In this study, the proposed EDPFA was used to modify the network parameters, which had the benefits of a high diagnostic accuracy and fast convergence.

This study applied the Taguchi method to implement the distributed parallel firefly algorithm (DPFA) and proposed an enhanced distributed parallel firefly algorithm (EDPFA). Then, this study aimed to use the proposed EDPFA in transformer fault diagnosis. Compared with other research, the main work of this study was as follows:

1. The distributed parallel firefly algorithm (DPFA) was implemented and then a new enhanced distributed parallel firefly algorithm (EDPFA) based on the Taguchi method was proposed.
2. The Taguchi method selected the better dimensions of different solutions to obtain a new solution, which was used as a new communication strategy for EDPFA.
3. The proposed EDPFA was tested by using the CEC2013 suite and had better performance than the standard FA and DPFA.
4. The proposed EDPFA was used to train the parameters of the BP neural network and improve the accuracy of the transformer fault diagnosis model based on the BP neural network.

The rest of the paper is structured as follows. Section 2 describes the original DPFA and the Taguchi method. Section 3 introduces the Taguchi method into the original DPFA and analyses the details of algorithm improvements. Section 4 focuses on testing the proposed EDPFA under the CEC2013 suite and compares it with other algorithms. Section 5 implements the proposed EDPFA in the field of transformer fault diagnosis. Section 6 sums up this paper.

2. Distributed Parallel Firefly Algorithm and Taguchi Method

This section provides a brief introduction to the original DPFA and Taguchi method.

2.1. Distributed Parallel Firefly Algorithm

The distributed parallel firefly algorithm (DPFA) was proposed by Pan and his associates in 2021 [24]. The DPFA is an updated version of the firefly algorithm (FA) proposed in 2007 [15]. The core idea of the DPFA is that the initial solutions are divided into some subgroups and share the information based on different communication strategies among subgroups after some fixed number of iterations.

2.1.1. The Mathematical Form of the DPFA

The search process of FA relates to two significant concepts: attractiveness and brightness. The attractiveness exists between two fireflies and indicates the position movement relationship between fireflies. The brightness is an individual characteristic of fireflies and is proportional to the fitness function. The standard FA satisfies the following three characteristics [15]: (1) Suppose that all fireflies can attract each other. (2) Fireflies' attractiveness is only related to distance and brightness. A firefly with a strong brightness will attract a firefly with a weak brightness. (3) The fitness function determines the brightness.

The mathematical form of the DPFA is as follows:

$$\beta(r) = \beta_0 e^{-\gamma\, r_{ij}^2} \tag{1}$$

$$r_{ij} = \left|\left|x_{i,g} - x_{j,g}\right|\right| = \sqrt{\sum_{k=1}^{d}\left(x_{i,g,k} - x_{j,g,k}\right)^2} \tag{2}$$

$$x_{i,g}(t+1) = x_{i,g}(t) + \beta\left(x_{j,g}(t) - x_{i,g}(t)\right) + \alpha\left(rand - \frac{1}{2}\right) \tag{3}$$

In Formula (1), $\beta(r)$ represents the attractiveness of two fireflies. β_0 represents the maximum degree of attractiveness ($r = 0$). Because the brightness will gradually weaken with the increase of distance and the absorption of the medium, the brightness absorption coefficient (γ) can be set as a constant to reflect the above characteristics.

In Formula (2), r_{ij} is the Cartesian distance between two fireflies. $x_{i,g}$ is the i th firefly in the g group. $x_{i,g,k}$ is the k th component of the spatial coordinate of firefly $x_{i,g}$.

In formula (3), the value of $x_{i,g}$ represents the brightness of firefly $x_{i,g}$. t represents the current iteration. $i = 1, 2, 3, \ldots, N_g$; $j = 1, 2, 3, \ldots, N_g$. N_g represents the number of fireflies in group g.

2.1.2. Communication Strategies

In the DPFA, when $t = nR$ ($n = 1, 2, 3, \ldots$), these subgroups trigger communication strategies. t and R represent the current iteration and the fixed communication iteration, respectively. The DPFA has four communication strategies, namely, the maximum of the same subgroup, the average of the same subgroup, the maximum of different subgroups and the average of different subgroups. The core ideal of communication strategies is to select some better solutions to replace the poorer ones in the subgroups. Different communication strategies have different ways of selecting better solutions. Take the maximum of the same subgroup as an example:

In strategy 1, when $t = nR$ iterations ($n = 1, 2, 3, \ldots$), the brightest firefly $x_{max,g}(t)$ in the same group will replace the darkest k fireflies in the same group. Figure 1 shows strategy 1.

$$x_{max,g}(t) = Max\{x_{1,g}(t), x_{2,g}(t), \ldots, x_{n,g}(t)\} \tag{4}$$

where $x_{1,g}(t), x_{2,g}(t), \ldots, x_{n,g}(t)$ represent all fireflies' positions in the gth group.

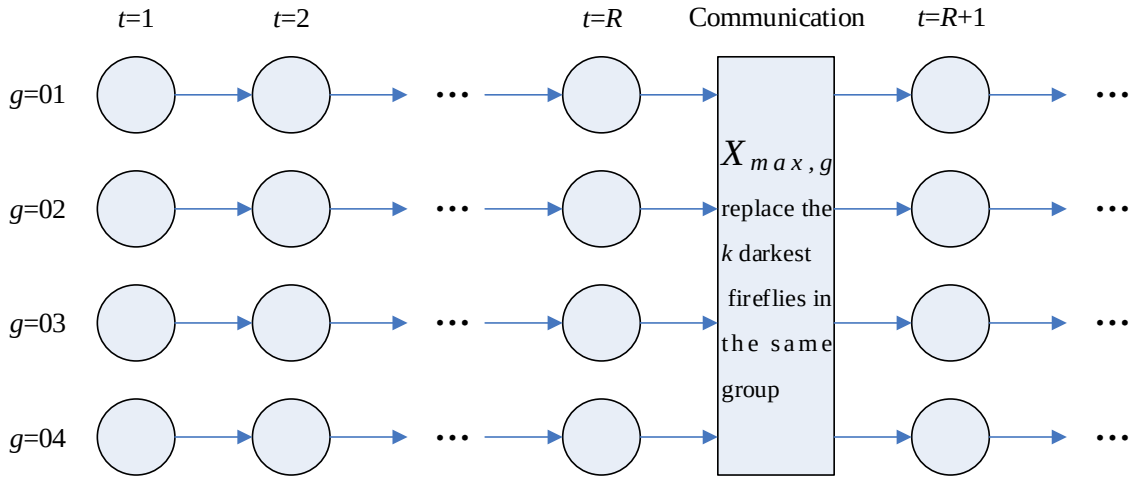


Figure 1. Strategy 1.

The other three communication strategies are as follows. More details of the DPFA are described in the literature [24].

Strategy 2: The average of the same subgroup:

$$x_{avg,g}(t) = \frac{x_{1,g}(t) + x_{2,g}(t) + x_{3,g}(t) + \ldots + x_{k,g}(t)}{k} \tag{5}$$

where $x_{1,g}(t), x_{2,g}(t), x_{3,g}(t), \ldots, x_{k,g}(t)$ represent the k brightest fireflies' positions in the gth group.

Strategy 3: The maximum of different subgroups:

$$x_{max}(t) = Max\{x_1(t), x_2(t), \ldots, x_N(t)\} \tag{6}$$

where $x_1(t), x_2(t), \ldots, x_N(t)$ represent all fireflies' positions in all groups.

Strategy 4: The average of different subgroups:

$$x_{avg}(t) = \frac{x_{max,1}(t) + x_{max,2}(t) + x_{max,3}(t) + \ldots + x_{max,G}(t)}{G} \tag{7}$$

where $x_{max,1}(t), x_{max,2}(t), x_{max,3}(t), \ldots, x_{max,G}(t)$ represent the brightest fireflies' positions in all groups.

For more detail on the DPFA, please refer to [24]. Algorithms 1 shows the pseudocode of DPFA.

Algorithms 1: The pseudo code of the DPFA.

Initialize the N fireflies and divide them evenly into G groups.
while $T < F$ do
for g = 1:G do
  Calculate the light intensity I_{ig} at x_{ig} using $f\left(x_{ig}\right)$ and rank the fireflies
  for i = 1:N/G do
    for j = 1:i do
      if ($I_{jg} > I_{ig}$)
      Move firefly i toward j in the g th subgroup in all D dimensions by using Equation (3)
      end if
    Evaluate distance r and update attractiveness.
    end for
  end for
if $T = nR$
  Communication strategies: apply $x_{max,g}$; $x_{avg,g}$; x_{max}; x_{avg} to update x_{ig} in all subgroups.
end if
$T = T + 1$
end while
Output :
The global best firefly x_{gbest} and the value of $f\left(x_{gbest}\right)$.

2.2. The Taguchi Method

The Taguchi method includes two major tools: (1) orthogonal arrays and (2) the signal-to-noise ratio (SNR) [10]. In the following, the concepts of these two tools are reviewed.

An array is said to be orthogonal if it satisfies two conditions: (1) each column represents a different level value of a considered factor and these considered factors can be evaluated independently, and (2) each row represents a set of parameters for an experiment. The orthogonal array can be described as

$$L_M\left(Q^K\right) \tag{8}$$

where K represents the number of columns (factors) and K is a positive integer. Q represents the number of level values of a considered factor, where Q is also a positive integer. M represents the number of experiments, where $M = K * (Q - 1) + 1$.

For instance, suppose that there are three sets of solutions with four parameters in an experiment. This means that each of the four factors can be at three levels. Then, Table 1 shows the orthogonal array $L_9\left(3^4\right)$. In the absence of the orthogonal array, if one wishes to find the optimal combination of parameters, the total number of experiments is $3^4 = 81$. However, orthogonal arrays provide us with a set of just nine experiments. The orthogonal array proposed by [12] can effectively reduce the number of experiments in the instance of obtaining the optimal combination of parameters.

Table 1. The orthogonal array $L_9\left(3^4\right)$.

Number of Experiments	Considered Factors			
	A	**B**	**C**	**D**
1	1	1	1	1
2	1	2	2	2
3	1	3	3	3
4	2	1	2	3
5	2	2	3	1
6	2	3	1	2
7	3	1	3	2
8	3	2	1	3
9	3	3	2	1

The SNR tool is used to find the parameters' optimal combination from all the combinations listed. To be more specific, the SNR is used to determine the appropriate level for each factor. The SNR can be calculated in various ways. For optimization problems, the value of the objective function can generally be regarded as the SNR.

3. Enhanced DPFA and Communication Strategy

In the original DPFA, the four communication strategies improve the algorithm through the group optimal solution or the global optimal solution, which has a great influence on the performance of the algorithm [24]. However, these strategies ignore the influence of various dimensions (parameters) in the optimal solution. Therefore, this study extracted all the dimensions (parameters) in the optimal solution and then used the Taguchi method to recombine the dimensions (parameters) to obtain a better solution.

3.1. Operation Strategy of the Taguchi Method

The operation strategy of the Taguchi method is described as follows:

Step 1: Choose k sets of solutions, which are denoted using the symbols $x_{1,g,d}$, $x_{2,g,d}, \ldots, x_{k,g,d}$. g represents the gth group and d represents the dth dimension of the solution space ($d = 1, 2, 3 \ldots, D$). D represents the total number of dimensions of the solution space.

Step 2: Each dimension of candidate solutions corresponds to a factor (the number of factors is D). The different values of candidate solutions denote different level values (the number of level values is k). The value of the objective function corresponding to each candidate solution is used as an SNR to judge whether the solution is good or bad. Next, it can combine these dimensions into a better solution (x_{better}) using the Taguchi method.

Step 3: The better solution (x_{better}) replaces the worst solution in the original groups.

To facilitate the reader's understanding, the following example is given.

Given the objective function $f(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2$, minimize it. Assume three solutions: $x_1 = [1, 2, 3, 0]$; $x_2 = [2, 0, 4, 3]$; $x_3 = [3, 3, 0, 2]$. $f(x_1) = 14$; $f(x_2) = 29$; $f(x_3) = 22$. Using the Taguchi method to combine these three solutions to get a better solution, Table 2 shows the results of solution combinations. According to Table 2, the best combination is $x_{better} = [2, 0, 0, 0]$, $f(x_{better}) = 4$.

Table 2. The results of solution combinations.

Experiment Number	Dimensions				*f(x)*
	d_1	d_2	d_3	d_4	
1	1	2	3	0	14
2	1	0	4	3	26
3	1	3	0	2	14
4	2	2	4	2	28
5	2	0	0	0	4
6	2	3	3	3	31
7	3	2	0	3	22
8	3	0	3	2	22
9	3	3	4	0	34

3.2. New Communication Strategies

In the original DPFA, the communication strategies are divided into two ways: intra-group information exchange (strategies 1 and 2) and inter-group information exchange (strategies 3 and 4). If the parameters of the solutions are independent, it is easier to obtain better results with the former. If the parameters of the solutions are loosely correlated, it is easier to obtain better results with the latter [38]. To improve the efficiency of information exchange, the Taguchi method is used to enhance the original communication strategies.

3.2.1. New Strategy 1

New strategy 1 follows the three steps in Section 3.1. The candidate solutions are the best k solutions in the group. New strategy 1 is an enhanced version of strategies 1 and 2 in the original DPFA. Figure 2 shows the new communication strategy 1.

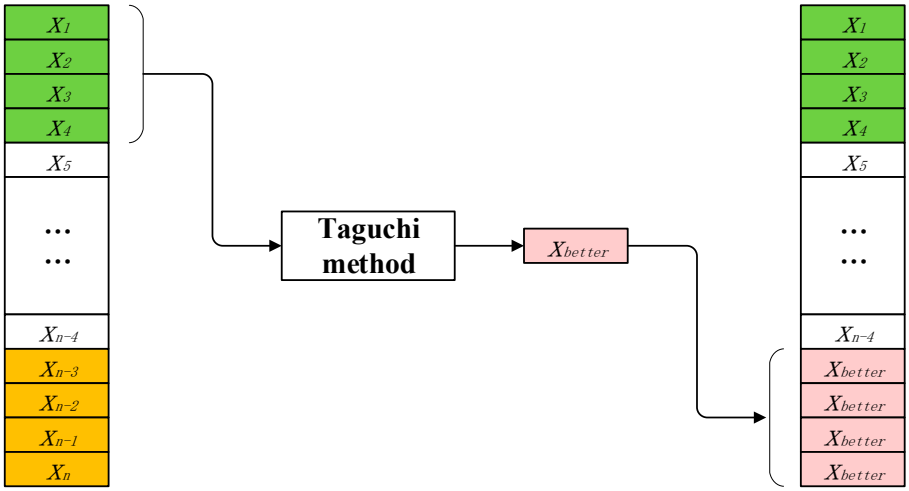


Figure 2. The new communication strategy 1.

3.2.2. New Strategy 2

New strategy 2 also follows the three steps in Section 3.1. The candidate solutions are the best solution in each group. New strategy 2 is an enhanced version of strategies 3 and 4 in the original DPFA. Figure 3 shows the new communication strategy 2.

3.3. The Pseudocode of the EDPFA

In the EDPFA, all initial solutions are divided into g subgroups. After the fixed iterations, these subgroups use the new communication strategy 1 or 2 to achieve the benefit of intra-group and inter-group collaboration. Algorithms 2 shows the pseudocode of the EDPFA.

Algorithms 2: The pseudocode of EDPFA.

Objective function $f(x)$, $x = (x_1, x_2, \ldots, x_d)$;
Initializing a population of N fireflies, $x_i (i \leq n)$;
Set the number of groups G.
while $t <$ *Max Generation*
for g = 1:G
  Calculate the light intensity I_{ig} using $f\left(x_{i,g}\right)$ and rank the fireflies.
  for i = 1:N/G
    for j = 1:i
      if $(I_{j,g} > I_{i,g})$
        Move firefly i toward j in the gth subgroup in all D dimensions by using Equation (3).
      end if
    Evaluate distance r by Equation (2) and update attractiveness using Equation (1).
    end for
  end for
end for
if $t = nR$
  Communication strategies: apply x_{better} to update the worst solutions.
end if
$t = t + 1$
end while

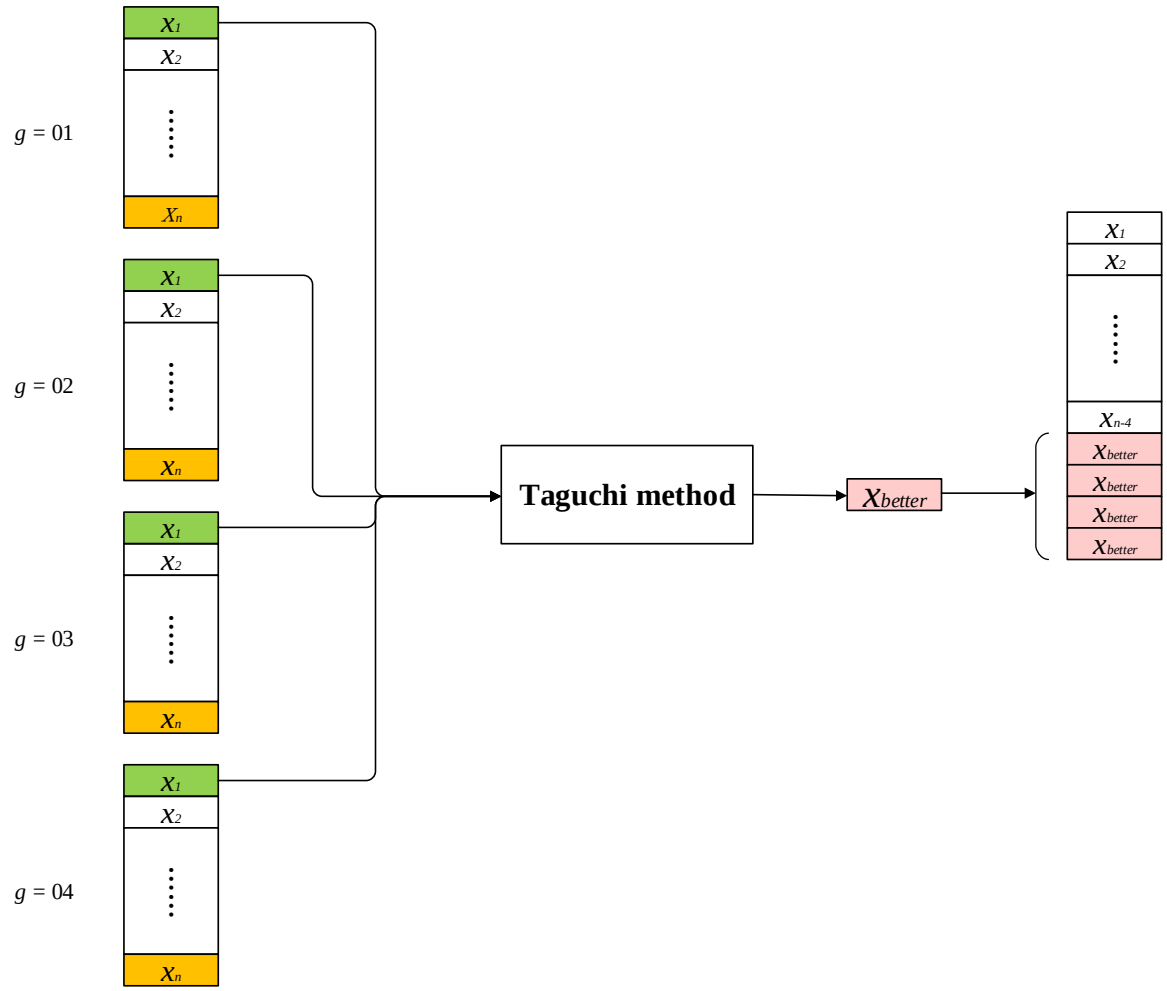


Figure 3. The new communication strategy 2.

4. Experiment Using the EDPFA

4.1. Test Functions and Parameters Setting

This study chose the CEC2013 suite to test the proposed EDPFA. The CEC2013 suite included unimodal functions ($f_1 \sim f_5$), multimodal functions ($f_6 \sim f_{20}$) and composite functions ($f_{21} \sim f_{28}$), and their dimensions were set to 30. The search range was set to $[-100, 100]$. More details of CEC2013 are presented in [39,40].

This study compared the proposed EDPFA with the FA and DPFA for testing the performance of algorithms. To assure the fairness of the experiment, 28 test functions were evaluated with 51 runs and 500 iterations. Because the operation of the Taguchi method calls test functions, the population size of the EDPFA was set to 94. Furthermore, the population size of the FA and DPFA was set to 100. In the experimental comparison, the number of function calls for all algorithms was the same. In addition, the three algorithms maintained consistent parameter settings ($\alpha = 0.25, \beta = 0.2, \gamma = 1,\ G = 4$). The programming was based on MATLAB 2019a. All the simulations were performed on a laptop with an AMD Ryzen7 2.90 GHz CPU and 16 GB RAM.

4.2. Comparison with the Original FA and DPFA

Table 3 shows the performance comparison results of the FA, DPFA and EDPFA from the "Mean" of 51 runs. The smaller the "Mean", the better the final result. The experimental results of FA and DPFA on each test function were compared with the EDPFA. The symbol (=) represents that the performance of the two algorithms was similar. The symbol (>) represents that the EDPFA performed well. The symbol (<) represents that

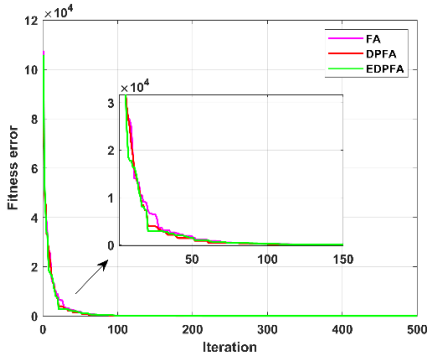
the EDPFA performed poorly. Finally, the last row of Table 3 counts the results on all benchmark functions.

Table 3. Comparison of the EDPFA with the FA and DPFA.

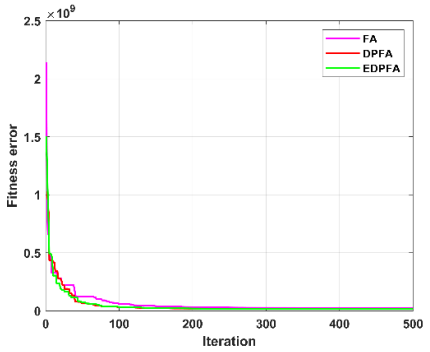
Function	FA		DPFA		EDPFA
f_1	5.60×10^{-3}	$>$	7.46×10^{-4}	$<$	2.50×10^{-3}
f_2	2.49×10^{7}	$>$	1.49×10^{7}	$<$	1.72×10^{7}
f_3	2.08×10^{9}	$>$	1.26×10^{8}	$<$	1.39×10^{8}
f_4	6.08×10^{4}	$>$	4.54×10^{4}	$<$	6.02×10^{4}
f_5	9.11×10	$>$	6.81×10	$<$	7.99×10
f_6	6.92×10	$>$	4.99×10	$<$	5.06×10
f_7	7.34×10	$>$	4.33×10	$>$	3.56×10
f_8	2.10×10	$=$	2.12×10	$>$	2.10×10
f_9	2.18×10	$>$	1.49×10	$>$	1.43×10
f_{10}	5.07	$>$	1.48	$>$	1.47
f_{11}	4.96×10	$>$	4.45×10	$>$	2.86×10
f_{12}	4.40×10	$>$	4.34×10	$>$	1.73×10
f_{13}	1.36×10^{2}	$>$	1.04×10^{2}	$>$	5.78×10
f_{14}	3.65×10^{3}	$>$	3.71×10^{3}	$>$	3.63×10^{3}
f_{15}	3.42×10^{3}	$>$	3.12×10^{3}	$>$	2.65×10^{3}
f_{16}	9.70×10^{-1}	$>$	1.73	$>$	3.48×10^{-1}
f_{17}	6.41×10	$>$	1.03×10^{2}	$>$	4.12×10
f_{18}	7.96×10	$>$	9.38×10	$>$	5.21×10
f_{19}	3.99	$>$	6.36	$>$	3.75
f_{20}	1.48×10	$=$	1.48×10	$=$	1.48×10
f_{21}	3.45×10^{2}	$>$	3.29×10^{2}	$>$	3.26×10^{2}
f_{22}	5.12×10^{3}	$>$	3.92×10^{3}	$<$	4.40×10^{3}
f_{23}	5.70×10^{3}	$>$	4.90×10^{3}	$>$	4.76×10^{3}
f_{24}	2.27×10^{2}	$>$	2.10×10^{2}	$>$	2.03×10^{2}
f_{25}	2.65×10^{2}	$>$	2.22×10^{2}	$>$	2.18×10^{2}
f_{26}	2.77×10^{2}	$<$	2.57×10^{2}	$<$	2.99×10^{2}
f_{27}	6.03×10^{2}	$>$	4.47×10^{2}	$>$	3.99×10^{2}
f_{28}	2.83×10^{2}	$<$	3.01×10^{2}	$>$	2.90×10^{2}
$\langle / = / \rangle$	24/2/2		19/1/8		-

According to the experimental results in Table 3, compared with the FA, the proposed EDPFA had 22 better results, 2 similar results and 2 bad results in 28 test functions. This result shows that EDPFA had a competitive search ability and solution accuracy. Compared with the DPFA, the proposed EDPFA had 19 better results, 1 similar result and 8 bad results in all test functions. This showed that the EDPFA was stronger than the EPFA in performance, and the DPFA was enhanced by the Taguchi method. However, regarding the results for test functions $f_1 \sim f_5$, the proposed EDPFA was not as good as the DPFA. $f_1 \sim f_5$ were the unimodal functions. The comparison results showed that the EDPFA was not suitable for solving the unimodal functions.

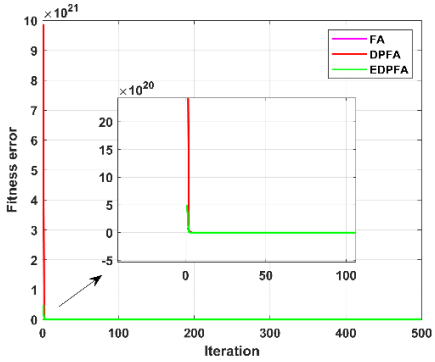
Next, to further evaluate the performances of the algorithms, the convergence curves of the FA, DPFA and EDPFA were compared. Each curve represented the convergence of the median value of the total 51 runs by a given algorithm, and some of them are presented in Figure 4. Table 4 summarizes the convergence figures under IEEE CEC 2013 for the 30D optimization. As shown in Figure 4, the proposed EDPFA could obtain a better convergence speed in some test functions (f_5, f_9, f_{11}, f_{12}, f_{13}, f_{14}, f_{15}, f_{16}, f_{17}, f_{21}, f_{22}, f_{23}, f_{24}, f_{25}, f_{27} and f_{28}). However, it did not have the best convergence speed in some test functions (f_4, f_6, f_8, f_{19}, f_{20} and f_{26}). Furthermore, in other test functions, the three algorithms had similar convergence performances. In general, compared with the FA and DPFA, the proposed EDFPA was more competitive in terms of convergence performance. In addition, the convergence effect of the FA was the worst out of the three algorithms.



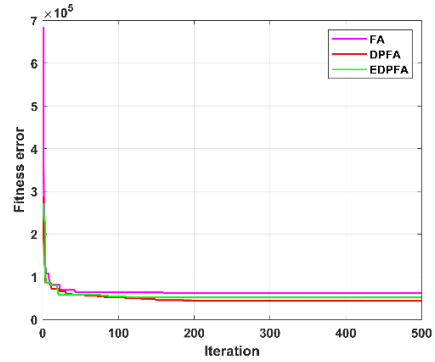
f_1



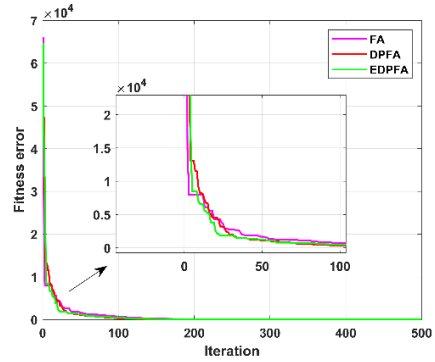
f_2



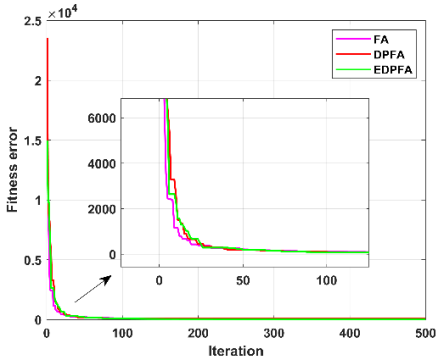
f_3



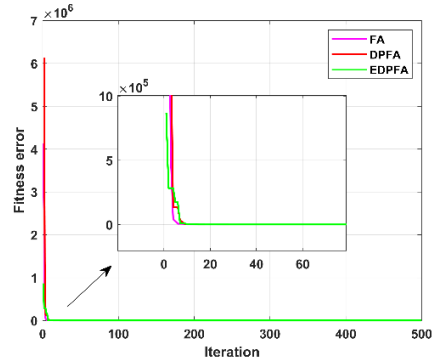
f_4



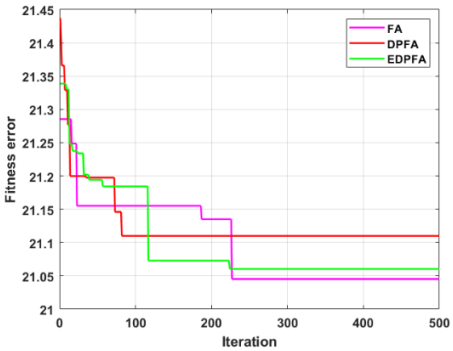
f_5



f_6

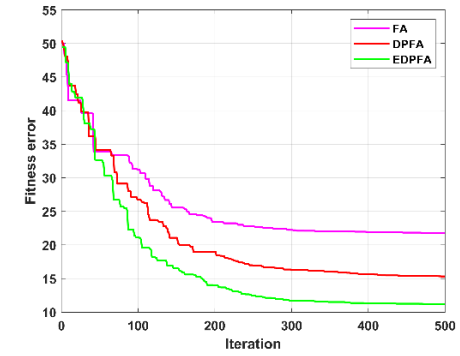


f_7

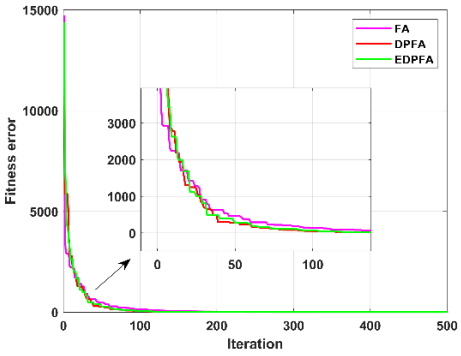


f_8

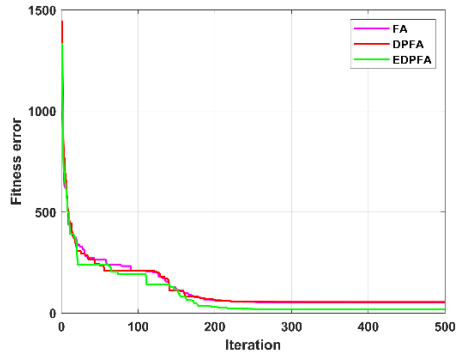
Figure 4. *Cont.*



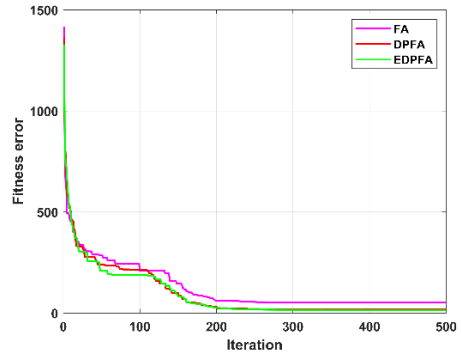
f_9



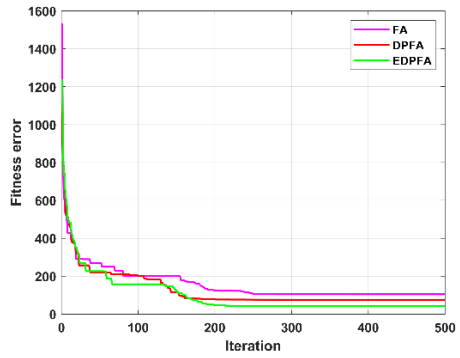
f_{10}



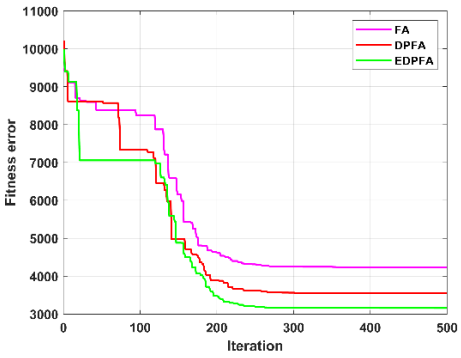
f_{11}



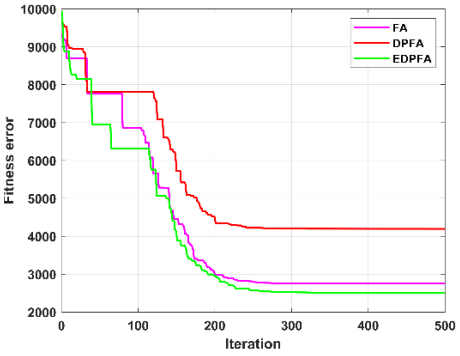
f_{12}



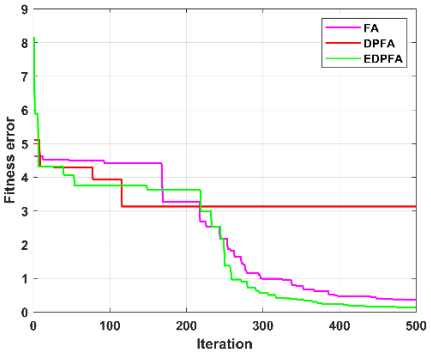
f_{13}



f_{14}

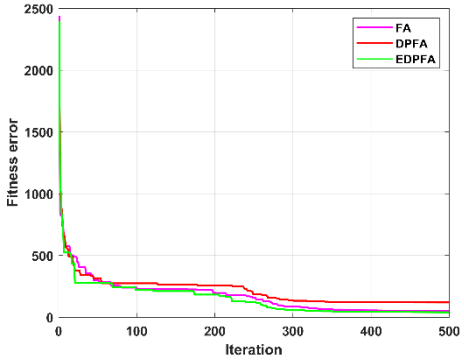


f_{15}

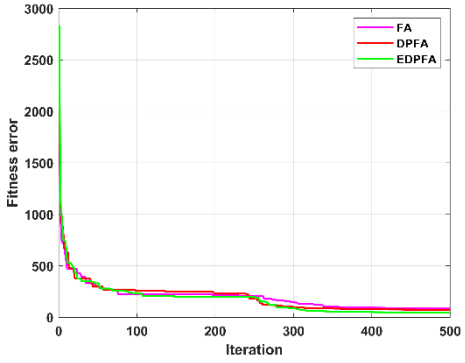


f_{16}

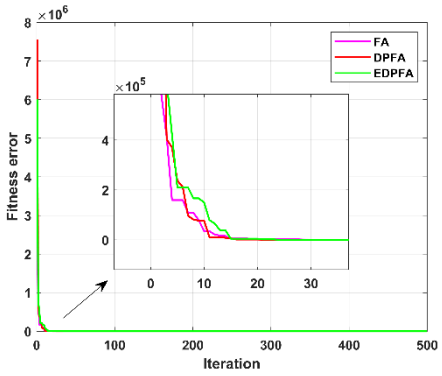
Figure 4. *Cont.*



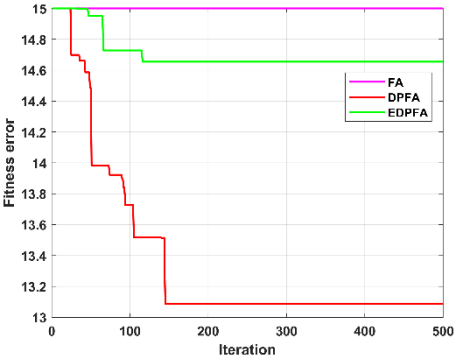
f_{17}



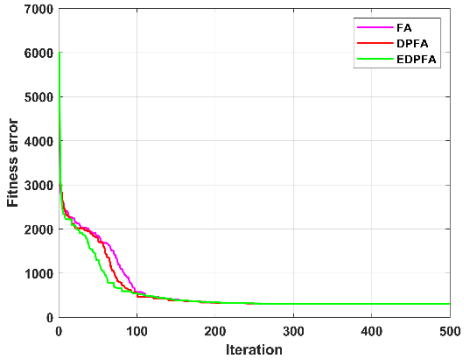
f_{18}



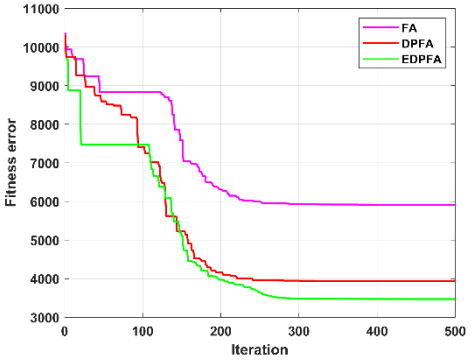
f_{19}



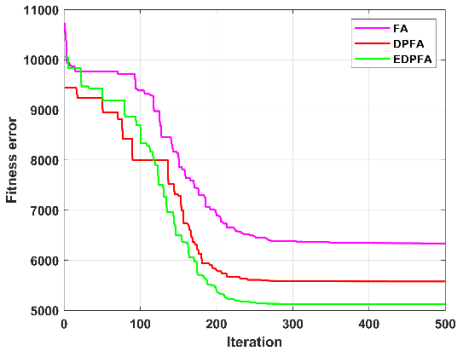
f_{20}



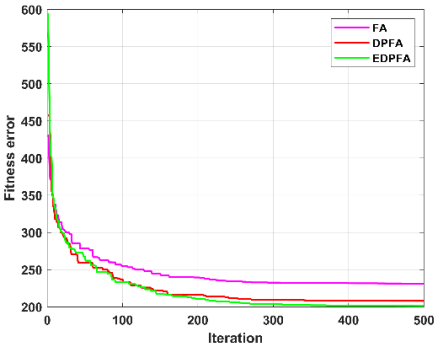
f_{21}



f_{22}



f_{23}



f_{24}

Figure 4. *Cont.*

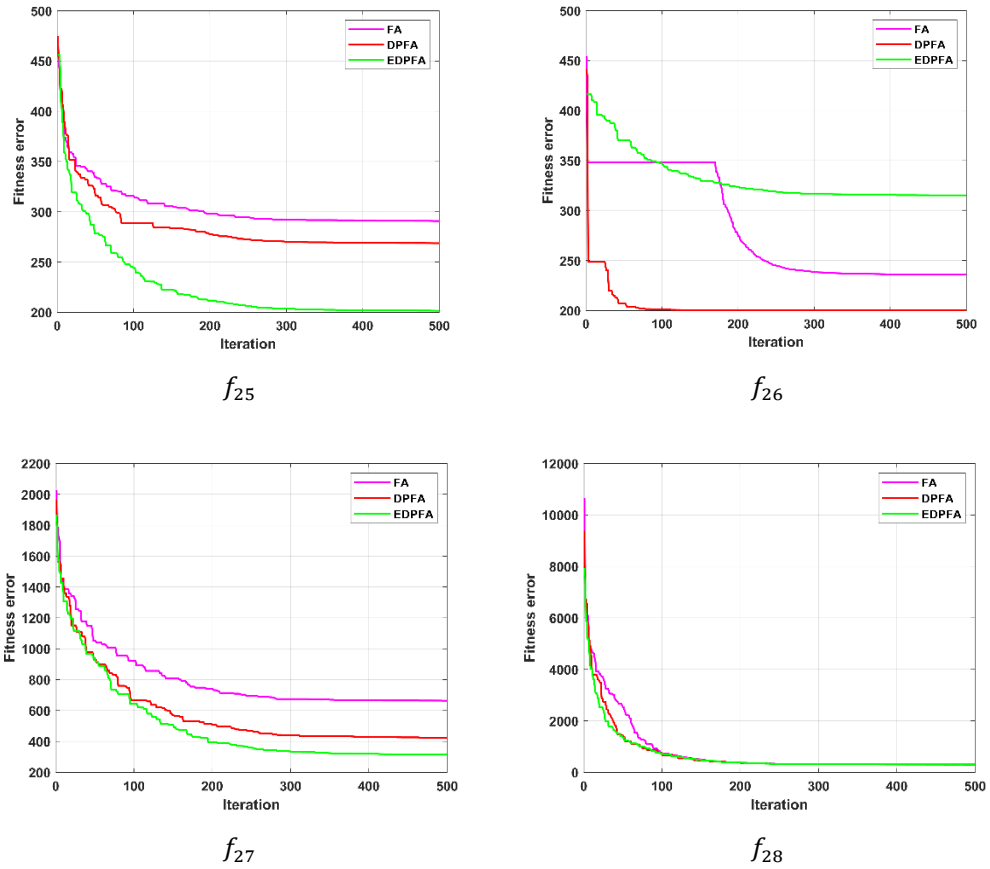


Figure 4. Convergence curves of the FA, DPFA and EDPFA.

Table 4. The summary of the convergence speed comparison of the FA, PCFA and DPCFA.

Name	Best (Tie Best) Performance	Similar Performance
FA	f_6, f_{19}	f_1, f_2, f_3, f_7, f_{10}, f_{18}
PCFA	f_4, f_8, f_{20}, f_{26}	
DPCFA	f_5, f_9, f_{11}, f_{12}, f_{13}, f_{14}, f_{15}, f_{16}, f_{17}, f_{21}, f_{22}, f_{23}, f_{24}, f_{25}, f_{27}, f_{28}	

4.3. Comparison with Other Algorithms

This section compares the performance of the EDPFA with some famous algorithms. All settings of the EDPFA were the same as in Sections 4.1 and 4.2.

Table 5 shows the performance comparison results of particle swarm optimization (PSO) [41], parallel particle swarm optimization (PPSO) [42], the genetic algorithm (GA) [11], the multi-verse optimizer (MVO) [43], the whole optimization algorithm (WOA) [44] and the ant lion optimizer (ALO) [45] in terms of the "Mean" of 51 runs. According to the data in Table 5, it is obvious that the proposed EDPFA performed better under the CEC2013 test suite. Compared with PSO, PPSO, the GA, the MVO, the WOA and the ALO, the proposed EDPFA achieved 24, 23, 24, 18, 26 and 21 better results, respectively.

Table 5. Comparison of the EDPFA with some common algorithms.

Function	PSO		PPSO		GA		EDPFA
f_1	9.56×10	$>$	4.96×10^{-1}	$>$	3.74×10^{-3}	$>$	2.50×10^{-3}
f_2	7.01×10^{6}	$<$	4.06×10^{6}	$<$	1.99×10^{7}	$>$	1.72×10^{7}
f_3	9.16×10^{9}	$>$	2.32×10^{9}	$>$	3.71×10^{8}	$>$	1.39×10^{8}
f_4	1.63×10^{4}	$<$	1.95×10^{4}	$<$	6.39×10^{4}	$>$	6.02×10^{4}

Table 5. *Cont.*

Function	PSO		PPSO		GA		EDPFA
f_5	1.09×10^2	$>$	4.19×10	$<$	4.11×10	$<$	7.99×10
f_6	9.24×10	$>$	7.51×10	$>$	7.42×10	$>$	5.06×10
f_7	1.37×10^2	$>$	1.06×10^2	$>$	4.37×10	$>$	3.56×10
f_8	2.10×10	$=$	2.09×10	$<$	2.11×10	$>$	2.10×10
f_9	3.25×10	$>$	3.16×10	$>$	1.32×10	$<$	1.43×10
f_{10}	7.23×10	$>$	6.43	$>$	1.82	$>$	1.47
f_{11}	2.66×10^2	$>$	2.23×10^2	$>$	3.09×10	$>$	2.86×10
f_{12}	2.61×10^2	$>$	2.13×10^2	$>$	4.28×10	$>$	1.73×10
f_{13}	3.66×10^2	$>$	3.16×10^2	$>$	5.52×10	$<$	5.78×10
f_{14}	4.43×10^3	$>$	4.15×10^3	$>$	4.49×10^3	$>$	3.63×10^3
f_{15}	4.35×10^3	$>$	3.97×10^3	$>$	3.05×10^3	$>$	2.65×10^3
f_{16}	1.30	$>$	1.24	$>$	2.45×10^{-1}	$<$	3.48×10^{-1}
f_{17}	2.70×10^2	$>$	1.96×10^2	$>$	4.56×10	$>$	4.12×10
f_{18}	2.48×10^2	$>$	1.93×10^2	$>$	7.15×10	$>$	5.21×10
f_{19}	2.17×10	$>$	1.22×10	$>$	3.99	$>$	3.75
f_{20}	1.45×10	$<$	1.46×10	$<$	1.50×10	$>$	1.48×10
f_{21}	3.76×10^2	$>$	3.56×10^2	$>$	3.03×10^2	$>$	3.26×10^2
f_{22}	5.26×10^3	$>$	4.91×10^3	$>$	5.43×10^3	$>$	4.40×10^3
f_{23}	5.52×10^3	$>$	5.19×10^3	$>$	5.27×10^3	$>$	4.76×10^3
f_{24}	3.00×10^2	$>$	2.92×10^2	$>$	2.32×10^2	$>$	2.03×10^2
f_{25}	3.36×10^2	$>$	3.26×10^2	$>$	2.78×10^2	$>$	2.18×10^2
f_{26}	3.10×10^2	$>$	3.00×10^2	$>$	3.33×10^2	$>$	2.99×10^2
f_{27}	1.17×10^3	$>$	1.16×10^3	$>$	4.87×10^2	$>$	3.99×10^2
f_{28}	2.73×10^3	$>$	1.93×10^3	$>$	3.02×10^2	$>$	2.90×10^2
$\langle / = / \rangle$	24/1/3		23/0/5		24/0/4		-
f_1	3.89×10^{-1}	$>$	2.08×10^2	$>$	1.49×10^{-5}	$<$	2.50×10^{-3}
f_2	7.64×10^6	$<$	7.28×10^7	$>$	1.52×10^7	$<$	1.72×10^7
f_3	4.97×10^8	$>$	3.04×10^{10}	$>$	1.03×10^9	$>$	1.39×10^8
f_4	3.45×10^3	$<$	9.02×10^4	$>$	8.17×10^4	$>$	6.02×10^4
f_5	7.22	$<$	4.56×10^2	$>$	6.04×10	$<$	7.99×10
f_6	3.68×10	$<$	1.84×10^2	$>$	6.77×10	$>$	5.06×10
f_7	6.97×10	$>$	4.47×10^2	$>$	1.38×10^2	$>$	3.56×10
f_8	2.10×10	$=$	2.10×10	$=$	2.10×10	$=$	2.10×10
f_9	2.01×10	$>$	3.90×10	$>$	3.06×10	$>$	1.43×10
f_{10}	1.92	$>$	5.17×10^2	$>$	8.39	$>$	1.47
f_{11}	1.01×10^2	$>$	5.07×10^2	$>$	2.44×10^2	$>$	2.86×10
f_{12}	1.06×10^2	$>$	5.36×10^2	$>$	2.03×10^2	$>$	1.73×10
f_{13}	1.94×10^2	$>$	6.53×10^2	$>$	3.19×10^2	$>$	5.78×10
f_{14}	3.66×10^3	$>$	5.95×10^3	$>$	4.60×10^3	$>$	3.63×10^3
f_{15}	3.75×10^3	$>$	5.53×10^3	$>$	4.41×10^3	$>$	2.65×10^3
f_{16}	1.29	$>$	1.54	$>$	1.08	$>$	3.48×10^{-1}
f_{17}	2.10×10^2	$>$	7.03×10^2	$>$	3.00×10^2	$>$	4.12×10
f_{18}	2.07×10^2	$>$	6.70×10^2	$>$	2.38×10^2	$>$	5.21×10
f_{19}	1.09×10	$>$	1.03×10^2	$>$	1.90×10	$>$	3.75
f_{20}	1.39×10	$<$	1.48×10	$=$	1.44×10	$<$	1.48×10
f_{21}	3.02×10^2	$<$	5.17×10^2	$>$	2.89×10^2	$<$	3.26×10^2
f_{22}	4.13×10^3	$<$	6.73×10^3	$>$	5.50×10^3	$>$	4.40×10^3
f_{23}	4.10×10^3	$<$	7.16×10^3	$>$	4.64×10^3	$<$	4.76×10^3
f_{24}	2.59×10^2	$>$	3.16×10^2	$>$	2.79×10^2	$>$	2.03×10^2
f_{25}	2.73×10^2	$>$	3.24×10^2	$>$	3.32×10^2	$>$	2.18×10^2
f_{26}	2.95×10^2	$<$	4.00×10^2	$>$	3.40×10^2	$>$	2.99×10^2
f_{27}	8.19×10^2	$>$	1.36×10^3	$>$	1.08×10^3	$>$	3.99×10^2
f_{28}	3.45×10^2	$>$	4.76×10^3	$>$	1.18×10^3	$>$	2.90×10^2
$\langle / = / \rangle$	18/1/9		26/2/0		21/1/6		-

5. Application for Transformer Fault Diagnosis

In machine learning, the backpropagation (BP) neural network has a strong ability to fit nonlinear systems. It is very suitable for solving prediction and classification problems [46]. Transformer fault diagnosis is essentially a fault classification problem. Therefore, it has been a research hotspot to introduce a BP neural network into the field of transformer fault diagnosis [47–50]. As described in this section, the proposed EDPFA was used to train the initial parameters of a BP neural network to improve the performance of transformer fault diagnosis model based on a BP neural network.

5.1. Structure of Transformer Fault Diagnosis Model Based on a BP Neural Network

The steps to establish the transformer fault diagnosis model based on a BP neural network were as follows:

Step 1: First, the characteristic gas content of transformers and the corresponding fault were composed into a data set.

Step 2: Then, 80% of the samples in the data set were used to train the BP neural network model. The other 20% of samples in the data set were used to test the trained BP neural network model.

Step 3: Finally, the transformer fault classification accuracy of the test set was counted to judge the performance of the model.

The transformer fault diagnosis data for dissolved gas in oil mainly include five fault gases (H_2, CH_4, C_2H_2, C_2H_4, C_2H_6) and their corresponding six fault types (normal state, NS; low-energy discharge, LED; arc discharge, AD; middle-and-low-temperature overheating, MLTO; high-temperature overheating, HTO; partial discharge, PD). Figure 5 shows the transformer fault diagnosis model based on BP neural network.

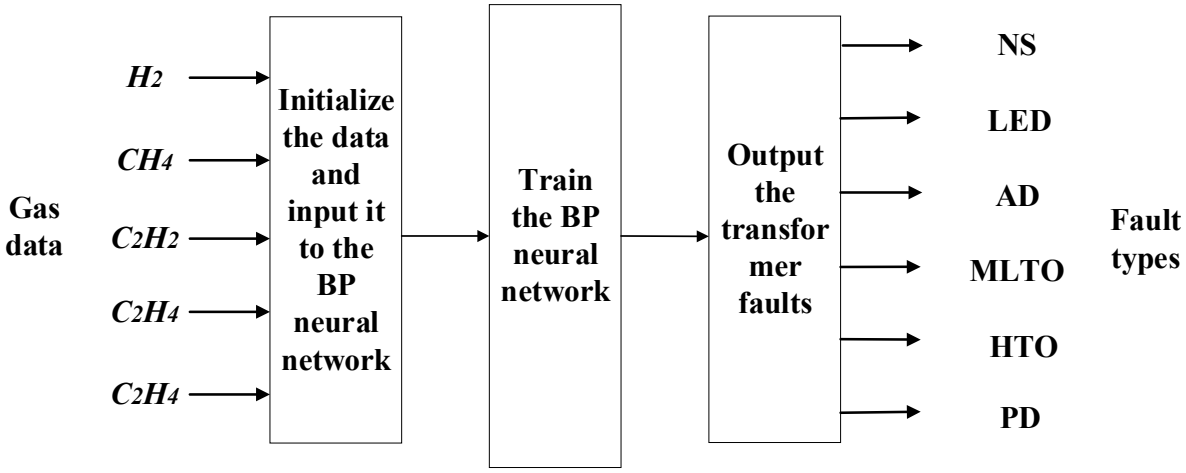


Figure 5. The transformer fault diagnosis model based on a BP neural network.

5.2. Structure of Transformer Fault Diagnosis Model Based on EDPFA-BP Neural Network

Even though the fitting ability of a traditional BP neural network is very strong, it still has some inherent defects, including low accuracy and slow convergence, which can no longer meet the requirements of a power system regarding transformer reliability [33]. The main reason is that all the thresholds and weights are randomly generated before the training of a BP neural network. These unoptimized initial values often lead to slow convergence and low accuracy of fault diagnosis results. Therefore, this study adopted the EDPFA to optimize the initial value of the BP neural network to improve the performance of the model. Figure 6 shows the transformer fault diagnosis model based on the EDPFA-BP neural network.

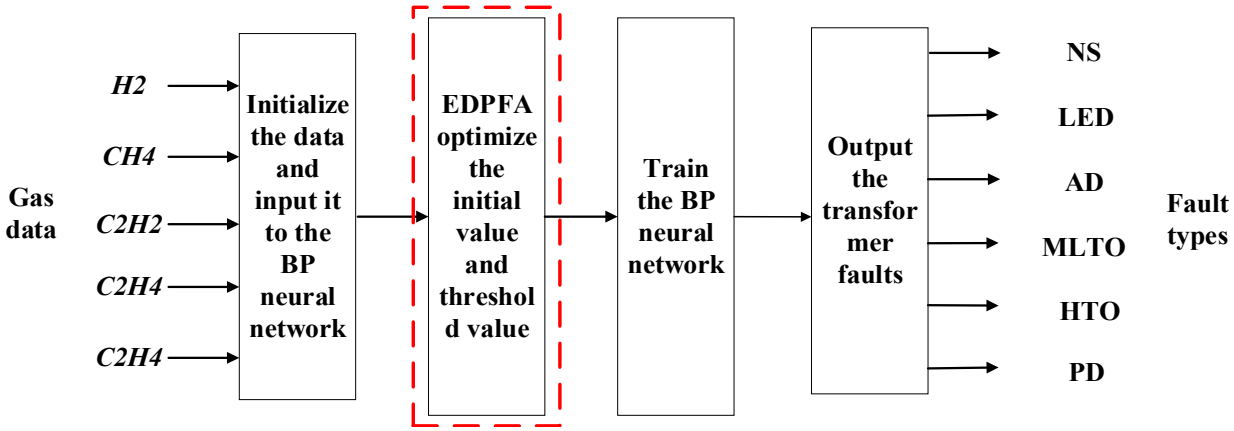


Figure 6. The transformer fault diagnosis model based on the EDPFA-BP neural network.

5.3. Experiment Process and Analysis

5.3.1. The Data Collection and Pretreatment

In this study, there were 465 sets of transformer fault data (including labels and features), some of which are shown in Table 6. Table 7 shows the codes of the transformer fault types. Figure 7 shows the sample distribution of the transformer fault types, in which the HTO faults had the highest number and the PD faults had the lowest number. To verify the model, 80% of the data of each fault type was randomly selected as the training set and 20% as the test set. In total, there were 375 sets of training data and 90 sets of testing data.

Table 6. Partial sample set of transformer fault diagnosis.

H_2	CH_4	C_2H_6	C_2H_4	C_2H_2	Fault Types
14.67	3.68	10.54	2.71	0.2	NS
7.5	5.7	3.4	2.6	3.2	NS
220	340	42	480	14	NS
30	110	137	52	22.3	NS
80	10	4	1.5	0	NS
46.13	11.57	33.14	8.52	0.63	NS
...	...				
345	112.25	27.5	51.5	58.75	LED
565	93	34	47	0	LED
550	53	34	20	0	LED
115.9	75	14.7	25.3	6.8	LED
78	161	86	353	10	LED
54	7	7.4	8.6	5.4	LED
...	...				
217.5	40	4.9	51.8	67.5	AD
1678	652.9	80.7	1005.9	419.1	AD
673.6	423.5	77.5	988.9	344.4	AD
60	40	6.9	110	70	AD
200	48	14	117	131	AD
46	37.2	8.3	107	71.9	AD
...	...				
181	262	210	528	0	MLTO
160	130	33	96	0	MLTO
4.32	193	118	125	0	MLTO
170	320	53	520	3.2	MLTO
27	90	42	63	0.2	MLTO
9259	8397	26,782	10,497	−1	MLTO
...	...				

Table 6. *Cont.*

H_2	CH_4	C_2H_6	C_2H_4	C_2H_2	Fault Types
172.9	334.1	172.9	812.5	37.7	HTO
25.1	411.91	320.9	1832.8	18.4	HTO
56	286	96	928	7	HTO
274	376	55	1002	17	HTO
15	12	5.3	3.2	0.2	HTO
56	285	96	28	7	HTO
...	...				
980	73	58	12	0	PD
650	53	34	20	0	PD
1565	93	34	47	0	PD
24.32	16.36	1.67	30.18	27.47	PD
2587.2	7.882	4.704	1.4	0	PD
980	73	58	12	0	PD

Table 7. The codes of transformer fault types.

Fault Types	NS	LED	AD	MLTO	HTO	PD
Code	01	02	03	04	05	06

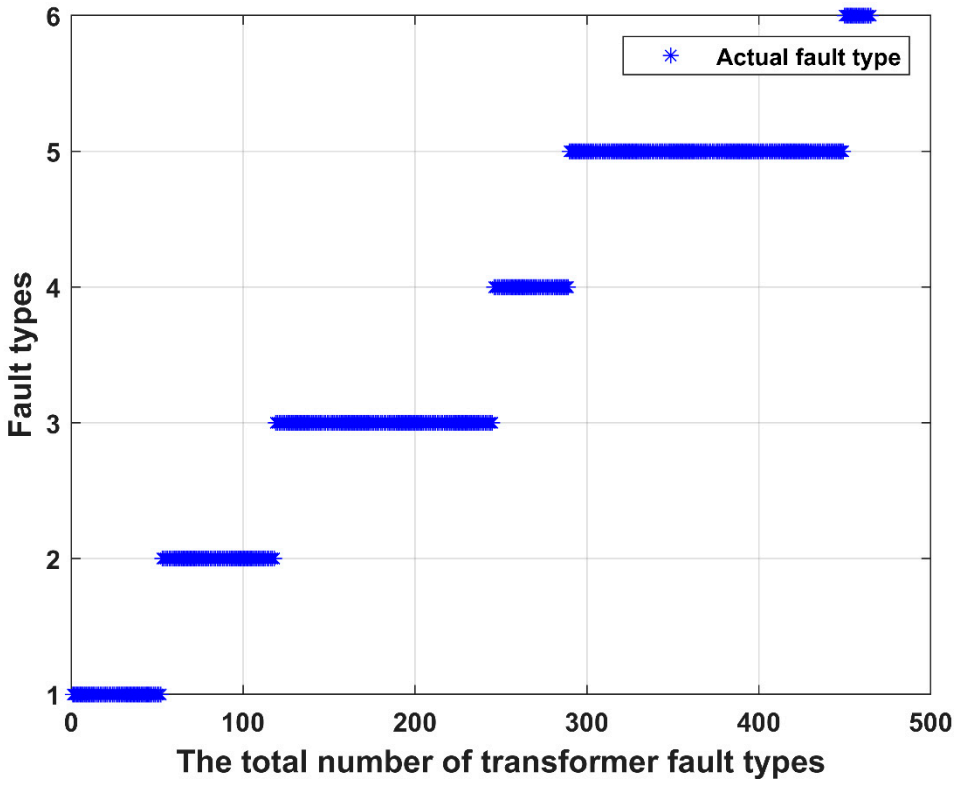


Figure 7. The sample distribution of the transformer fault types.

5.3.2. The Parameter Setting of a BP Neural Network

A BP neural network is a kind of mathematical model that can simulate complex nonlinear relations and automatically modify parameters. In a BP neural network, there are input layers, hidden layers and output layers. The signal first travels through the input layer, then to the hidden layer and finally to the output layer. In the above process, the relevant information is processed by regulating internal relations between lots of nodes.

Figure 8 shows the topological type of the BP neural network adopted in this study. The number of inputs was 5 (five fault gases), the number of hidden layers was 12, the number

of output layers was 6 and the number of outputs was 6 (six fault types). In addition, after many experimental trials, this study set the iteration times and learning precision goal of the BP neural network as 1000 and 0.0001, respectively. The activation function adopts a sigmoid function and the BP neural network introduced error backpropagation into the multilayer networks.

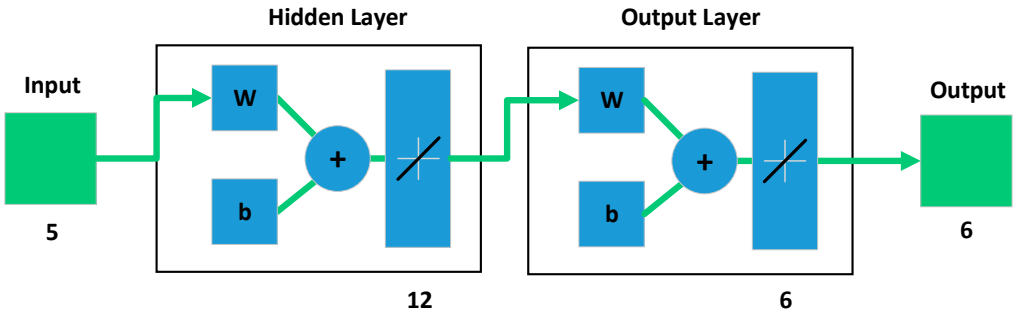


Figure 8. The topological type of the BP neural network.

To ensure the objectivity of the experiment process, all parameters in each transformer fault diagnosis model were the same. The parameters to be used for the EDPFA were consistent with Section 4.

5.3.3. Experiment Results and Analysis

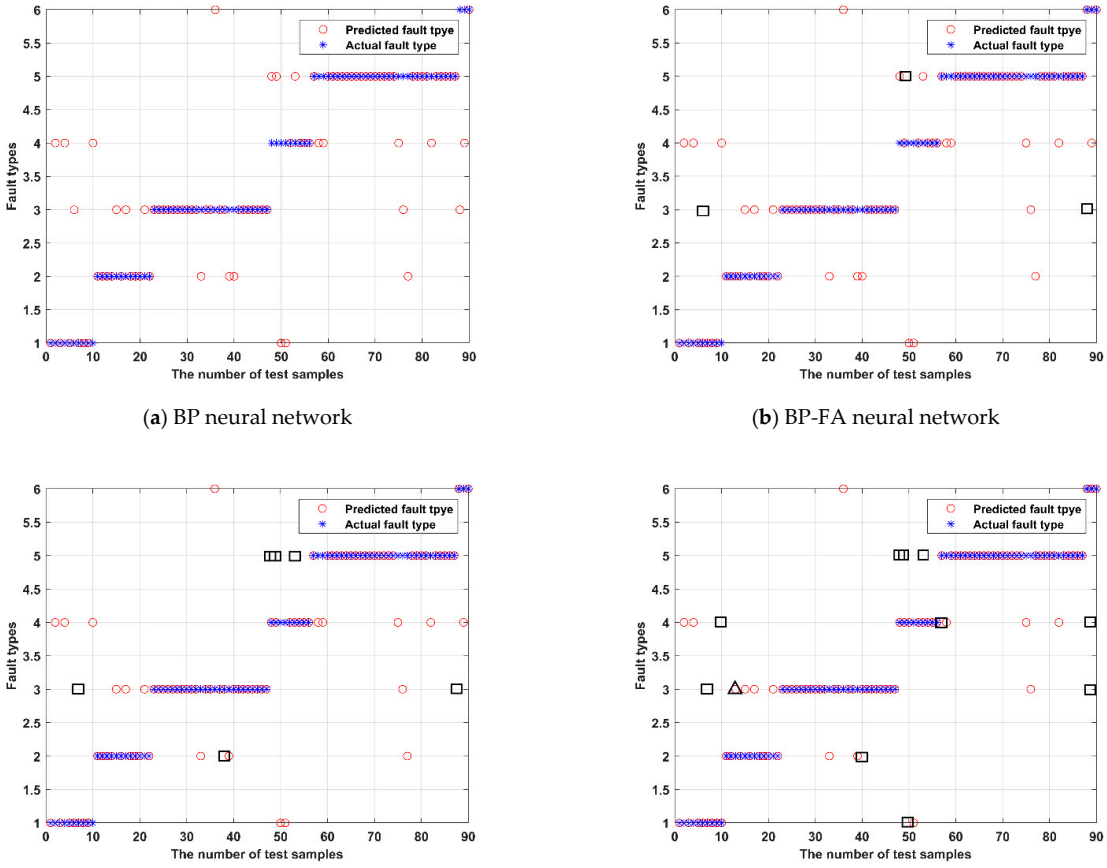
Figure 9 shows the diagnosis results, which included four models (the BP neural network, FA-BP neural network, DPFA-BP neural network and EDPFA neural network). In Figure 9, the ordinate represents the six transformer fault types, and the abscise represents 465 sets of transformer faults. The "○" in Figure 9 represents a predicted fault type, and the "✱" represents an actual fault type. If the "○" and the "✱" overlap, this transformer fault was correctly predicted; otherwise, the prediction was wrong. To make the results more intuitive, "□" represents the improved BP neural network identifies correctly, while the original BP neural network made an incorrect identification. "△" is the opposite. Table 8 shows the diagnosis accuracy of each model.

Table 8. The accuracy of each model.

Transformer Fault Diagnosis Model	Accuracy (%)
BP neural network	73.33%
FA-BP neural network	76.67%
DPFA-BP neural network	80.00%
EDPFA-BP neural network	84.44%

As shown in Figure 9, compared with other models based on the improved BP neural network (b–d), there were more "○" faults in the unimproved BP model (a). This shows that the transformer fault diagnosis models based on the BP neural network had poor fault classification ability. Furthermore, it is obvious that, compared with other neural networks, the EDPFA-BP neural network had better performance regarding fault 4 (middle-and-low-temperature overheating), where it identified fault 4 more often. From Table 8, the fault classification accuracy of the BP-EDPFA neural network was the highest (up to 84.44%). Compared with the other models, the accuracy of EDPFA-BP neural network was higher by 11.11%, 6.66% and 3.34%. The recall and precision of each model are shown in Tables 9 and 10. As shown in Table 9, the EDPFA-BP neural network had the highest recall rate for six fault types. Especially regarding the PD fault, its recall rate reached 100%. From Table 10, the precision of the BP neural network was the lowest, and the precision of the EDPFA-BP neural network was the highest. This indicated that the EDPFA-BP neural network had a better classification effect and fewer fault classification errors. From the above three aspects, it can be concluded that the proposed EDPFA could better optimize

the initial parameters of the BP neural network and manage the transformer fault diagnosis model based on a BP neural network.



(**a**) BP neural network

(**b**) BP-FA neural network

(**c**) BP-DPFA neural network

(**d**) BP-EDPFA neural network

Figure 9. The diagnosis results from the four models.

Table 9. The recall of each model.

Recall (%)						
Transformer Fault Diagnosis Model	**NS**	**LED**	**AD**	**MLTO**	**HTO**	**PD**
BP neural network	60.00%	75.00%	84.00%	44.44%	80.65%	33.33%
FA-BP neural network	70.00%	75.00%	84.00%	55.56%	80.65%	66.66%
DPFA-BP neural network	70.00%	75.00%	88.00%	77.78%	80.65%	66.66%
EDPFA-BP neural network	80.00%	66.67%	88.00%	88.89%	87.10%	100%

Table 10. The precision of each model.

Precision (%)						
Transformer Fault Diagnosis Model	**NS**	**LED**	**AD**	**MLTO**	**HTO**	**PD**
BP neural network	75.00%	69.23%	77.78%	33.33%	89.29%	50%
FA-BP neural network	77.78%	69.23%	84.00%	38.46%	92.59%	66.66%
DPFA-BP neural network	77.78%	75.00%	84.62%	46.67%	100%	66.66%
EDPFA-BP neural network	88.89%	80.00%	81.48%	61.54%	100%	75.00%

6. Conclusions

An enhanced distributed parallel firefly algorithm (DEPFA) based on the Taguchi method was proposed and it was applied to transformer fault diagnosis. The Taguchi method could be used to improve the effectiveness of the original communication strategies in the DPFA, which enhanced the influence of various dimensions (parameters) in the optimal solution. In the test functions, the implemented EDPFA achieved faster convergence and could find better solutions. Compared with the FA and DPFA, the EDPFA had 24 and 19 better results. This is important for the safety and stability of a power system to quickly diagnose and predict the existing or latent transformer faults. The proposed EDPFA was used to train the BP neural network to implement diagnoses. The experimental results showed that the proposed EDPFA could effectively improve the accuracy of the transformer fault diagnosis model based on a BP neural network (up to 11.11%). However, the EDPFA is not fully studied and there is still a lot of room for optimization, especially regarding solving the unimodal optimization problems.

Author Contributions: Conceptualization, Z.-J.L. and W.-G.C.; methodology, J.S.; software, J.S.; validation, Z.-J.L.; formal analysis, Z.-Y.Y.; investigation, L.-Y.C.; resources, Z.-J.L.; data curation, Z.-J.L.; writing—original draft preparation, J.S.; writing—review and editing, Z.-J.L. All authors have read and agreed to the published version of the manuscript.

Funding: State Grid Corporation of China Science and Technology Project [5500-202099279A-0-0-00].

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

References

1. Fister, I., Jr.; Yang, X.S.; Fister, I.; Brest, J.; Fister, D. A brief review of nature-inspired algorithms for optimization. *arXiv* **2013**, arXiv:1307.4186.
2. Blum, C.; Li, X. Swarm intelligence in optimization. In *Swarm Intelligence*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 43–85.
3. Shang, J.; Tian, Y.; Liu, Y.; Liu, R. Production scheduling optimization method based on hybrid particle swarm optimization algorithm. *J. Intell. Fuzzy Syst.* **2018**, *34*, 955–964. [CrossRef]
4. Osorio, C.; Chong, L. A computationally efficient simulation-based optimization algorithm for large-scale urban transportation problems. *Transp. Sci.* **2015**, *49*, 623–636. [CrossRef]
5. Xu, X.; Rong, H.; Trovati, M.; Liptrott, M.; Bessis, N. CS-PSO: Chaotic particle swarm optimization algorithm for solving combinatorial optimization problems. *Soft Comput.* **2018**, *22*, 783–795. [CrossRef]
6. Wang, X.; Pan, J.S.; Chu, S.C. A parallel multi-verse optimizer for application in multilevel image segmentation. *IEEE Access* **2020**, *8*, 32018–32030. [CrossRef]
7. Pan, J.S.; Shan, J.; Zheng, S.G.; Chu, S.C.; Chang, C.K. Wind power prediction based on neural network with optimization of adaptive multi-group salp swarm algorithm. *Clust. Comput.* **2021**, *24*, 2083–2098. [CrossRef]
8. Abido, M.A. Optimal design of power-system stabilizers using particle swarm optimization. *IEEE Trans. Energy Convers.* **2002**, *17*, 406–413. [CrossRef]
9. Karna, S.K.; Sahai, R. An overview on Taguchi method. *Int. J. Eng. Math. Sci.* **2012**, *1*, 1–7.
10. Roy, R.K. *A Primer on the Taguchi Method*; Society of Manufacturing Engineers: Southfield, MI, USA, 2010.
11. Labani, M.; Moradi, P.; Jalili, M. A multi-objective genetic algorithm for text feature selection using the relative discriminative criterion. *Expert Syst. Appl.* **2020**, *149*, 113276. [CrossRef]
12. Saucedo-Dorantes, J.J.; Jaen-Cuellar, A.Y.; Delgado-Prieto, M.; de Jesus Romero-Troncoso, R.; Osornio-Rios, R.A. Condition monitoring strategy based on an optimized selection of high-dimensional set of hybrid features to diagnose and detect multiple and combined faults in an induction motor. *Measurement* **2021**, *178*, 109404. [CrossRef]
13. Lin, H.L.; Chou, C.P. Optimization of the GTA welding process using combination of the Taguchi method and a neural-genetic approach. *Mater. Manuf. Process.* **2010**, *25*, 631–636. [CrossRef]
14. Tsai, P.W.; Pan, J.S.; Chen, S.M.; Liao, B.Y. Enhanced parallel cat swarm optimization based on the Taguchi method. *Expert Syst. Appl.* **2012**, *39*, 6309–6319. [CrossRef]
15. Subbaraj, P.; Rengaraj, R.; Salivahanan, S. Enhancement of self-adaptive real-coded genetic algorithm using Taguchi method for economic dispatch problem. *Appl. Soft Comput.* **2011**, *11*, 83–92. [CrossRef]
16. Yang, X.S. Firefly algorithms for multimodal optimization. In *International Symposium on Stochastic Algorithms*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 169–178.
17. Yang, X.S.; He, X. Firefly algorithm: Recent advances and applications. *Int. J. Swarm Intell.* **2013**, *1*, 36–50. [CrossRef]
18. Xue, X. A compact firefly algorithm for matching biomedical ontologies. *Knowl. Inf. Syst.* **2020**, *62*, 2855–2871. [CrossRef]

19. Shan, J.; Chu, S.C.; Weng, S.W.; Pan, J.S.; Jiang, S.J.; Zheng, S.G. A parallel compact firefly algorithm for the control of variable pitch wind turbine. *Eng. Appl. Artif. Intell.* **2022**, *111*, 104787. [CrossRef]
20. Farahani, S.M.; Abshouri, A.A.; Nasiri, B.; Meybodi, M. A Gaussian firefly algorithm. *Int. J. Mach. Learn. Comput.* **2011**, *1*, 448. [CrossRef]
21. Gandomi, A.H.; Yang, X.S.; Talatahari, S.; Alavi, A.H. Firefly algorithm with chaos. *Commun. Nonlinear Sci. Numer. Simul.* **2013**, *18*, 89–98. [CrossRef]
22. Crawford, B.; Soto, R.; Suárez, M.O.; Paredes, F.; Johnson, F. Binary firefly algorithm for the set covering problem. In Proceedings of the 2014 9th Iberian Conference on Information Systems and Technologies (CISTI), Barcelona, Spain, 18–21 June 2014; pp. 1–5.
23. Sai, V.O.; Shieh, C.S.; Nguyen, T.T.; Lin, Y.C.; Horng, M.F.; Le, Q.D. Parallel Firefly Algorithm for Localization Algorithm in Wireless Sensor Network. In Proceedings of the 2015 Third International Conference on Robot, Vision and Signal Processing (RVSP), Kaohsiung, Taiwan, 18–20 November 2015. [CrossRef]
24. Shan, J.; Pan, J.S.; Chang, C.K.; Chu, S.C.; Zheng, S.G. A distributed parallel firefly algorithm with communication strategies and its application for the control of variable pitch wind turbine. *ISA Trans.* **2021**, *115*, 79–94. [CrossRef]
25. Apostolopoulos, T.; Vlachos, A. Application of the firefly algorithm for solving the economic emissions load dispatch problem. *Int. J. Comb.* **2010**, *2011*, 523806. [CrossRef]
26. Patle, B.K.; Parhi, D.R.; Jagadeesh, A.; Kashyap, S.K. On firefly algorithm: Optimization and application in mobile robot navigation. *World J. Eng.* **2017**, *14*, 65–76. [CrossRef]
27. Gokhale, S.S.; Kale, V.S. An application of a tent map initiated Chaotic Firefly algorithm for optimal overcurrent relay coordination. *Int. J. Electr. Power Energy Syst.* **2016**, *78*, 336–342. [CrossRef]
28. Mahdavi-Meymand, A.; Zounemat-Kermani, M. A new integrated model of the group method of data handling and the firefly algorithm (GMDH-FA): Application to aeration modelling on spillways. *Artif. Intell. Rev.* **2020**, *53*, 2549–2569. [CrossRef]
29. Wang, M.; Vandermaar, A.J.; Srivastava, K.D. Review of condition assessment of power transformers in service. *IEEE Electr. Insul. Mag.* **2002**, *18*, 12–25. [CrossRef]
30. Zhang, D.; Li, C.; Shahidehpour, M.; Wu, Q.; Zhou, B.; Zhang, C.; Huang, W. A bi-level machine learning method for fault diagnosis of oil-immersed transformers with feature explainability. *Int. J. Electr. Power Energy Syst.* **2022**, *134*, 107356. [CrossRef]
31. de Faria, H., Jr.; Costa, J.G.S.; Olivas, J.L.M. A review of monitoring methods for predictive maintenance of electric power transformers based on dissolved gas analysis. *Renew. Sustain. Energy Rev.* **2015**, *46*, 201–209. [CrossRef]
32. Zhang, Y.; Chen, E.; Guo, P.J.; Ma, C. Application of improved particle swarm optimization BP neural network in transformer fault diagnosis. In Proceedings of the 2017 Chinese Automation Congress (CAC), Jinan, China, 20–22 October 2017; pp. 6971–6975.
33. Benmahamed, Y.; Kherif, O.; Teguar, M.; Boubakeur, A.; Ghoneim, S.S. Accuracy improvement of transformer faults diagnostic based on DGA data using SVM-BA classifier. *Energies* **2021**, *14*, 2970. [CrossRef]
34. Arshad, M.; Islam, S.M.; Khaliq, A. Fuzzy logic approach in power transformers management and decision making. *IEEE Trans. Dielectr. Electr. Insul.* **2014**, *21*, 2343–2354. [CrossRef]
35. Fei, S.; Zhang, X. Fault diagnosis of power transformer based on support vector machine with genetic algorithm. *Expert Syst. Appl.* **2009**, *36*, 11352–11357. [CrossRef]
36. Wang, M.H.; Tseng, Y.F.; Chen, H.C.; Chao, K.H. A novel clustering algorithm based on the extension theory and genetic algorithm. *Expert Syst. Appl.* **2009**, *36*, 8269–8276. [CrossRef]
37. Ou, M.; Wei, H.; Zhang, Y.; Tan, J. A dynamic adam based deep neural network for fault diagnosis of oil-immersed power transformers. *Energies* **2019**, *12*, 995. [CrossRef]
38. Wang, R.-B.; Wang, W.-F.; Xu, L.; Pan, J.-S.; Chu, S.-C. An Adaptive Parallel Arithmetic Optimization Algorithm for Robot Path Planning. *J. Adv. Transp.* **2021**, *2021*, 1–22. [CrossRef]
39. Liang, J.J.; Qu, B.Y.; Suganthan, P.N.; Hernández-Díaz, A.G. *Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization*; Technical Report 201212; Computational Intelligence Laboratory, Zhengzhou University: Zhengzhou, China; Nanyang Technological University: Singapore, 2013; pp. 281–295.
40. Tanabe, R.; Fukunaga, A. Evaluating the performance of SHADE on CEC 2013 benchmark problems. In Proceedings of the 2013 IEEE Congress on Evolutionary Computation, Cancun, Mexico, 20–23 June 2013; pp. 1952–1959.
41. Marini, F.; Walczak, B. Particle swarm optimization (PSO). A tutorial. *Chemom. Intell. Lab. Syst.* **2015**, *149*, 153–165. [CrossRef]
42. Chu, S.C.; Roddick, J.F.; Pan, J.S. A parallel particle swarm optimization algorithm with communication strategies. *J. Inf. Sci. Eng.* **2005**, *21*, 809–818.
43. Aljarah, I.; Mafarja, M.; Heidari, A.A.; Faris, H.; Mirjalili, S. Multi-verse optimizer: Theory, literature review, and application in data clustering. In *Nature-Inspired Optimizers*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 123–141.
44. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
45. Mirjalili, S. The ant lion optimizer. *Adv. Eng. Softw.* **2015**, *83*, 80–98. [CrossRef]
46. Li, J.; Cheng, J.H.; Shi, J.Y.; Huang, F. Brief introduction of back propagation (BP) neural network algorithm and its improvement. In *Advances in Computer Science and Information Engineering*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 553–558.
47. Sun, Y.J.; Zhang, S.; Miao, C.X.; Li, J.M. Improved BP neural network for transformer fault diagnosis. *J. China Univ. Min. Technol.* **2007**, *17*, 138–142. [CrossRef]

48. Luo, Y.; Hou, Y.; Liu, G.; Tang, C. Transformer fault diagnosis method based on QIA optimization BP neural network. In Proceedings of the 2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, China, 15–17 December 2017; pp. 1623–1626.
49. Yan, C.; Li, M.; Liu, W. Transformer fault diagnosis based on BP-Adaboost and PNN series connection. *Math. Probl. Eng.* **2019**, *2019*, 1019845. [CrossRef]
50. Li, X.; Chen, Z.; Fan, X.; Xu, Q.; Lu, J.; He, T. Fault diagnosis of transformer based on BP neural network and ACS-SA. *High Volt. Appar.* **2018**, *54*, 134–139.