algorithms

Article

Locating the Parameters of RBF Networks Using a Hybrid Particle Swarm Optimization Method

Ioannis G. Tsoulos *,† and Vasileios Charilogis †

Department of Informatics and Telecommunications, University of Ioannina, 47150 Arta, Greece
* Correspondence: itsoulos@uoi.gr
† These authors contributed equally to this work.

Abstract: In the present work, an innovative two-phase method is presented for parameter tuning in radial basis function artificial neural networks. These kinds of machine learning models find application in many scientific fields in classification problems or in function regression. In the first phase, a technique based on particle swarm optimization is performed to locate a promising interval of values for the network parameters. Particle swarm optimization was used as it is a highly reliable method for global optimization problems, and in addition, it is one of the fastest and most-flexible techniques of its class. In the second phase, the network was trained within the optimal interval using a global optimization technique such as a genetic algorithm. Furthermore, in order to speed up the training of the network and due to the use of a two-stage method, parallel programming techniques were utilized. The new method was applied to a number of famous classification and regression datasets, and the results were more than promising.

Keywords: neural networks; particle swarm optimization; genetic algorithms

Citation: Tsoulos, I.G.; Charilogis V. Locating the Parameters of RBF Networks Using a Hybrid Particle Swarm Optimization Method. *Algorithms* **2023**, *16*, 71. https://doi.org/10.3390/a16020071

Academic Editors: Nebojsa Bacanin, Eva Tuba, Milan Tuba and Ivana Strumberger

Received: 12 December 2022
Revised: 16 January 2023
Accepted: 17 January 2023
Published: 21 January 2023

1. Introduction

Regression and data classification are two major categories of problems that are solved with machine learning techniques. Such problems appear regularly in scientific fields such as physics [1,2], chemistry [3,4], economics [5,6], medicine [7,8], etc. A programming tool that is used quite often to handle such problems is the Radial Basis Function (RBF) artificial neural network [9]. An RBF network can be defined as the following function:

$$y(\overrightarrow{x}) = \sum_{i=1}^{k} w_i \phi(\|\overrightarrow{x} - \overrightarrow{c_i}\|) \tag{1}$$

The following applies to the above equation:

1. The vector $\overrightarrow{x}$ stands for the input pattern to the Equation (1). The number of elements in this vector is denoted as d.
2. The vectors $\overrightarrow{c_i}$, $i = 1, .., k$ are denoted as the center vectors.
3. The vector $\overrightarrow{w}$ is considered as the output weight of the RBF network.
4. The value $y(\overrightarrow{x})$ represents the predicted value of the network for the pattern $\overrightarrow{x}$.

Typically, the Gaussian function can bed used as the function $\phi(x)$, and it is defined as:

$$\phi(x) = \exp\left(-\frac{(x-c)^2}{\sigma^2}\right) \tag{2}$$

A plot of the previous function with $c = 0$, $\sigma = 1$ is displayed in Figure 1. As can be observed, the value of the function decreases as we move away from the center. An extensive overview of RBF networks was given in the work of Ghosh and Nag [10]. RBF networks are used as approximation tools in various cases, such as solutions to differential equations [11,12], digital communications [13,14], physics [15,16], chemistry [17,18], economics [19–21], network security [22,23], etc. RBF networks were thoroughly discussed in [24], and they have been parallelized in a variety of research papers [25,26]. This model has been extended by various researchers in tasks such as creating new initialization techniques for the network parameters, [27–29], pruning techniques [30–32], the construction of RBF networks [33–35] etc.

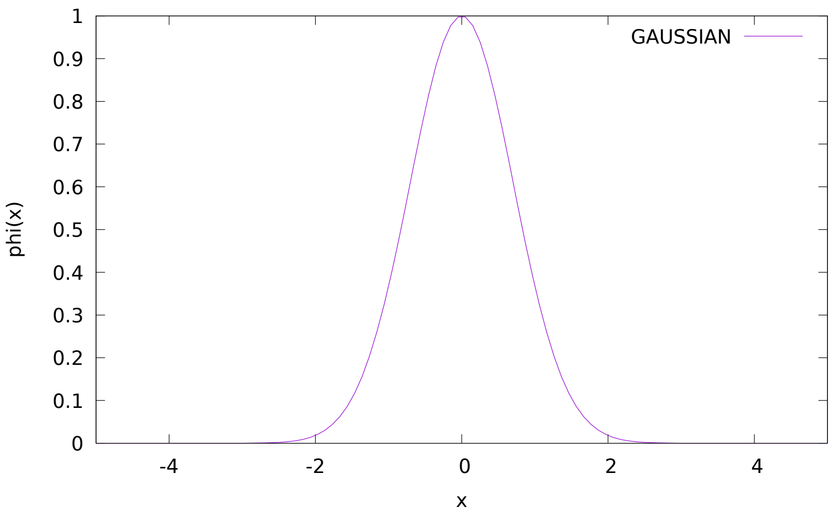

Figure 1. A typical plot for the Gaussian function, for $c = 0$ and $\sigma = 1$.

In this work, a hybrid technique is proposed for the optimal calculation of the parameters of an RBF network. This technique consists of two phases. During the first phase, information was collected from the training data of the neural network and an attempt was made to identify a small interval of values for the neural network parameters. To identify this interval, an optimization method was used, which gradually creates the optimal value interval, which was estimated to give the lowest value for the training error of the network. To locate the optimal interval, the Particle Swarm Optimization (PSO) technique was used [36]. The PSO method was chosen for the first phase because it is fast and flexible enough for optimization problems, does not require a large number of parameters to be input by the user, and has been successfully used in a variety of problems such as flow shop scheduling [37], developing charging strategies for electric vehicles [38], emotion recognition [39], robot trajectory planning [40], etc. The detection of the value interval was performed in order to then make the minimization of the network error faster and more efficient in the second phase of the optimization method. In the second phase, the parameters of the neural network were optimized within the optimal value interval of the first phase. The optimization can be performed by any global optimization method [41]. In this work, genetic algorithms [42–44] were chosen for the second phase. The main advantages of genetic algorithms are tolerance to errors, easy implementation in parallel, efficient exploration of the search space, etc.

Recently, much work has been appeared to tune the parameters of machine learning models, such as the work of Agarwal and Bhanot [45] for the adaptation of the RBF parameters, the incorporation of an improved ABC algorithm to adapt the parameters of RBF networks [46], the usage of the Firefly algorithm for optimization [47], along with machine learning models for cervical cancer diagnosis [48], the adaptation of the CNN and XGBOOST models by an optimization algorithm for COVID-19 diagnosis [49], etc.

The rest of this article is organized as follows: in Section 2, the two phases of the proposed method are thoroughly discussed; in Section 3, the experimental datasets are listed, as well as the experimental results; finally, in Section 4, some conclusions are presented.

2. Method Description

The training error of the RBF network is expressed as:

$$E(y(x,g)) = \sum_{i=1}^{m} (y(x_i,g) - t_i)^2 \tag{3}$$

The value m stands for the number of patterns, and the values t_i denote the real output for the input. The vector g denotes the set of parameters of the RBF network. Usually, RBF networks are trained through a two-phase procedure:

1. In the first phase, the k centers, as well as the associated variances are calculated through the K-means algorithm [50]. A typical formulation of the K-means algorithm is outlined in Algorithm 1.
2. In the second phase, the weight vector $\overrightarrow{w} = (w_1, w_2, \ldots, w_k)$ is estimated by solving a linear system of equations:
   (a) **Set** $W = w_{kj}$;
   (b) **Set** $\Phi = \phi_j(x_i)$;
   (c) **Set** $T = \{t_i = f(x_i), i = 1,.., M\}$**;**
   (d) The system to be solved is identified as:

$$\Phi^T \left(T - \Phi W^T \right) = 0 \tag{4}$$

   with the solution:

$$W^T = \left(\Phi^T \Phi \right)^{-1} \Phi^T T = \Phi^{\dagger} T. \tag{5}$$

The proposed work used two computational phases to optimally calculate the network parameters. Firstly, a promising range of the parameters of the network was calculated through an optimization process that incorporated interval arithmetic. Subsequently, the parameters of the network were trained with the usage of a genetic algorithm inside the located range of the first phase. The following subsections analyze both of these phases in detail.

Algorithm 1 The K-means algorithm.

```
1. Repeat.
   (a) $S_j = \{\},\ j = 1..k$.
   (b) For each pattern $x_i,\ i = 1, ..., m$, do:
       i.  Calculate $j^* = \min_{i=1}^{k}\{D(x_i, c_j)\}$.
       ii. Update $S_{j^*} = S_{j^*} \cup \{x_i\}$.
   (c) EndFor.
   (d) For each center $c_j,\ j = 1..k$, do:
       i.  Define $M_j$ as the number of points in $S_j$.
       ii. Calculate $c_j$:
           $c_j = \frac{1}{M_j} \sum_{i=1}^{M_j} x_i$
   (e) EndFor.
2. Compute the variances for every center as
   $\sigma_j^2 = \frac{\sum_{i=1}^{M_j} (x_i - c_j)^2}{M_j}$
3. Terminate if there is no change in centers $c_j$.
```

2.1. Preliminaries

In order to perform interval arithmetic on RBF networks, the following definitions are introduced:

1. The comparison of two intervals $W = [w_1, w_2]$, $Z = [z_1, z_2]$ is performed through the function:

$$L^*(W,Z) \quad = \quad \begin{cases} \text{TRUE}, & w_1 < z_1, \text{OR } (w_1 = z_1 \text{ AND } w_2 < z_2) \\ \text{FALSE}, & \text{OTHERWISE;} \end{cases} \tag{6}$$

2. The function $E(y)$ (Equation (3)) is modified to an interval one $\left[E_{\min}(y), E_{\max}(y)\right]$ calculated with the procedure given in Algorithm 2.

In the proposed algorithm, the RBF network contains n variables, where

$$n = (d+2) \times k \tag{7}$$

The value of n is calculated as follows:

1. Every center $\overrightarrow{c_i}$, $i = 1, .., k$ has d variables, which means $d \times k$ variables.
2. For every center, a separate value σ_i is used for the Gaussian processing unit, which means k variables.
3. The output weight $\overrightarrow{w}$ also has k variables.

Algorithm 2 Fitness calculation for the modified PSO algorithm.

The fitness calculation for a given particle g is as follows:

1. **Take** N_S random samples in g.
2. **Calculate** $E_{\min}(g) = \min_{g_i \in N_S}\left(\left(\sum_{j=1}^{M} y(x_j, g_i) - t_j\right)^2\right)$.
3. **Calculate** $E_{max}(g) = \max_{g_i \in N_S}\left(\left(\sum_{j=1}^{M} y(x_j, g_i) - y_j\right)^2\right)$.
4. **Return** $f_g = [E_{min}(g), E_{max}(g)]$.

2.2. The Proposed PSO Algorithm

During this phase, arithmetic interval techniques are used to locate a range for the parameters of the RBF network. The interval techniques [51–53] comprise a common method in global optimization with various applications [54–56]. The first phase aims to locate the most-promising bounding box for the n parameters of the corresponding RBF network. The initial bounding box is defined as S, which is a subset of $\mathbb{R}^n$:

$$S = [a_1, b_1] \otimes [a_2, b_2] \otimes \ldots [a_n, b_n] \tag{8}$$

The interval method of the first phase divides the set S subsequently by discarding areas that are not promising enough to contain the global minimum. In order to locate the best interval for Equation (8), a modified PSO algorithm [57] is used. The proposed variant of the PSO method is based on the original technique (Algorithm 1 of [57]); however, the particles are intervals of values, and at each iteration, a normalization of the velocity vector takes place to avoid generating particles outside the original range of values. The PSO method is based on a population of candidate solutions, which, in most cases, are called particles. The method is based on two vectors: the current location of particles denoted as $\overrightarrow{p}$ and the velocity of their movement denoted as $\overrightarrow{u}$. The PSO method finds the global minimum by moving the particles based on their previous best position, as well as the best position of the total population of particles.

The initial bounding boxes for the centers and variances of the RBF network are constructed using the K-means clustering algorithm. Subsequently, the initial values for the intervals $[a_i, b_i]$ are calculated through Algorithm 3. The values for the intervals of the first

$(d+1) \times k$ variables are obtained as a multiple of the positive quantity F with the values obtained by K-means. The value B is used to initialize the intervals for the output weight $\overrightarrow{w}$. Afterwards, the following PSO variant is executed:

1. **Set** N_c as the amount of particles.
2. **Set** the normalization factor λ.
3. **Set** the k weights of the RBF network.
4. **Set** N_g the maximum generations allowed.
5. **Set** N_s the number of random samples that will be used in the fitness calculation algorithm.
6. **Set** $f^* = [\infty, \infty]$, the fitness of the best located particle p^*.
7. **Construct** $S = [a_1, b_1] \otimes [a_2, b_2] \otimes \ldots [a_n, b_n]$, as obtained from the previous two algorithms.
8. **Initialize** the N_c particles. Each particle $p_i,\ i = 1, \ldots, N_c$ is considered as a set of intervals randomly initialized in S. The layout of each particle is graphically presented in Figure 2.
9. **For** $i = 1, \ldots, N_c$, **do:**
   (a) **Calculate** the fitness f_i of particle p_i using the procedure outlined in Algorithm 2.
   (b) **If** $L^*(f_i, f^*) =$ TRUE, **then** $f^* = f_i,\ p^* = p_i$
   (c) **Set** $p_{b,i} = p_i,\ f_{b,i} = f_i$ as the best located position for particle i and the associated fitness value.
   (d) **For** $j = 1, \ldots, n$, **do**:
      i. **Set** δ the width of interval p_{ij}.
      ii. **Set** $u_{ij} = \left[-r\frac{\delta}{20}, r\frac{\delta}{20}\right]$, with r a random number in $[0, 1]$. The velocity is initialized to a small sub-interval of the range of values for the corresponding parameter in order to avoid, as much as possible, excessive values for the velocity. This would result in the particles moving out of their value range very quickly, thus making the optimization process difficult.
   (e) **EndFor**.
10. **EndFor**.
11. **Set** iter = 0.
12. **Calculate** the inertia value as $\omega = \omega_{\max} - \frac{\text{iter}}{N_g}(\omega_{\max} - \omega_{\min})$, where common values for these parameters are $\omega_{\min} = 0.4$ and $\omega_{\max} = 0.9$. Many inertia calculations have appeared in the relevant literature such as constant inertia [58], linearly decreasing inertia [59], exponential inertia [60], random inertia calculation [61], dynamic inertia [62], fuzzy inertia calculation [63], etc. The present method of calculating the inertia was chosen because it decreases linearly with time, and for large values of the inertia, it allows a wider search in the search space, while for low values, it allows a more focused search.
13. **For** $i = 1, \ldots, N_c$, **do:**
   (a) **Calculate** the new velocity $u_i = \omega u_i + r_1 c_1 (p_{b,i} - p_i) + r_2 c_2 (p^* - p_i)$, where r_1, r_2 are random numbers in $[0, 1]$, and the constant values c_1 and c_2 stand for the cognitive and the social parameters, correspondingly. Usually, the values for c_1 and c_2 are in $[1, 2]$.
   (b) **Normalize** the velocity as: $u_i = \frac{1}{\lambda} u_i$, where λ is a positive number with $\lambda > 1$.
   (c) **Update** the position $p_i = p_i + u_i$.
   (d) **Calculate** the fitness f_i of particle p_i.
   (e) **If** $L^*(f_i, f_{b,i}) =$ TRUE, **then** $p_{b,i} = p_i,\ f_{b,i} = f_i$.
   (f) **If** $L^*(f_i, f^*) =$ TRUE, **then** $f^* = f_i,\ p^* = p_i$.
14. **EndFor**.
15. **Set** iter = iter+1.
16. **If** iter$\leq N_g$, goto Step 13.

```
17. Else, return $S = [a_1, b_1] \otimes [a_2, b_2] \otimes \ldots [a_n, b_n]$, the domain range for the best particle $p^*$.
```

Algorithm 3 Algorithm used to locate the initial values for $[a_i, b_i],\ i = 1, ..., n$.

```
1. Set m = 0.
2. Set $F > 1,\ B > 0$.
3. For $i = 1..k$, do:
    (a) For $j = 1..d$, do:
        i. Set $a_m = -F \times c_{ij}$, $b_m = F \times c_{ij}$.
        ii. Set $m = m + 1$.
    (b) EndFor.
    (c) Set $a_m = -F \times \sigma_i$, $b_m = F \times \sigma_i$.
    (d) Set $m = m + 1$.
4. EndFor.
5. For $j = 1, ..., k$, do:
    (a) Set $a_m = -B,\ b_m = B$.
    (b) Set $m = m + 1$.
6. EndFor.
```

c_{11}	c_{12}	...	c_{1d}	σ_1	c_{21}	c_{22}	...	c_{2d}	σ_2	...	c_{k1}	c_{k2}	...	c_{kd}	σ_k	w_1	w_2	$\ldots$	w_k

Figure 2. The scheme of the particles in the current PSO algorithm.

2.3. Optimization of Parameters through Genetic Algorithm

During the second phase of the proposed method, a genetic algorithm is implemented, which optimizes the parameters of the RBF network within the optimal interval calculated in the first phase. The used genetic algorithm has its roots in the GA(c_{r1}, l) algorithm from the paper of Kaelo and Ali [64]. This method was enhanced using the stopping rule suggested by Tsoulos [65]. This genetic algorithm has the following steps:

1. **Initialization step:**
    (a) **Set** N_c as the number of chromosomes. Every chromosome is coded as in the case of PSO using the scheme of Figure 2.
    (b) **Set** N_g as the maximum number of generations allowed.
    (c) **Set** k as the weight number of the RBF network.
    (d) **Obtain** the domain range S from the procedure of Section 2.2.
    (e) **Initialize** N_C randomly in S.
    (f) **Define** the selection rate $p_s \in [0, 1]$.
    (g) **Define** the mutation rate $p_m \in [0, 1]$.
    (h) **Set** iter = 0.
2. **Evaluation step:**
    For every chromosome g, **calculate** the associated fitness value $f_g = \sum_{i=1}^{m} (y(x_i, g) - t_i)^2$:
3. **Genetic operations step**:
    Perform the genetic operations of selection, crossover, and mutation.
    (a) **Selection procedure:** First, the population of chromosomes is sorted based on the associated fitness values. The first $(1 - p_s) \times N_c$ chromosomes are copied unchanged to the next generation, while the rest are replaced by offspring constructed by the crossover procedure. During the selection step, a series of mating pairs is chosen using the well-known procedure of tournament selection for each parent.

(b) **Crossover procedure**: For each pair (z, w) of chosen parents, two new offspring $\tilde{z}$ and $\tilde{w}$ are constructed with the steps:

$$\begin{aligned} \tilde{z}_i &= a_i z_i + (1 - a_i) w_i \\ \tilde{w}_i &= a_i w_i + (1 - a_i) z_i \end{aligned} \tag{9}$$

where a_i is a random number with $a_i \in [-0.5, 1.5]$ [64].

(c) **Mutation procedure**: For every element of each chromosome, pick a random number $r \in [0, 1]$. **If** $r \leq p_m$, then alter randomly the corresponding element.

4. **Termination check step:**
   (a) **Set** $iter = iter + 1$.
   (b) **If** the termination criteria hold, then **Terminate**; **else, goto** evaluation step.

The overall process of the two phases is graphically shown in Figure 3.

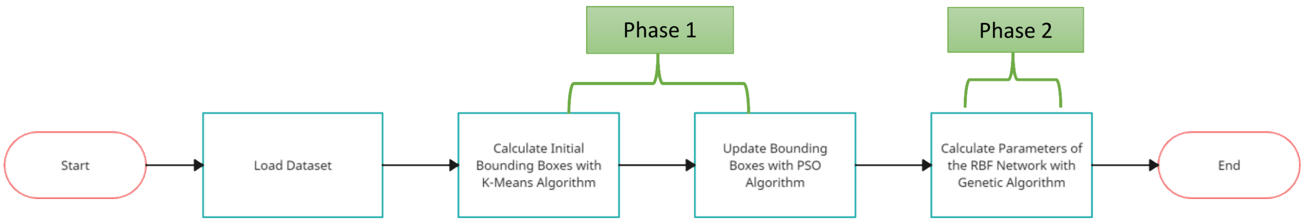
Phase 1
Phase 2
Start
Load Dataset
Calculate Initial Bounding Boxes with K-Means Algorithm
Update Bounding Boxes with PSO Algorithm
Calculate Parameters of the RBF Network with Genetic Algorithm
End

Figure 3. Graphical representation of the proposed two-phase method.

3. Experiments

The suggested method was tested on a series of classification and regression problems found from various papers and sites of the relevant literature. For the classification problems, two Internet databases were used:

1. The UCI dataset repository, https://archive.ics.uci.edu/ml/index.php (accessed on 5 January 2023).
2. The Keel repository, https://sci2s.ugr.es/keel/datasets.php (accessed on 5 January 2023) [66].

The regression problems can be found at the Statlib URL ftp://lib.stat.cmu.edu/datasets/index.html (accessed on 5 January 2023).

3.1. Experimental Datasets

The classification problems used here were the following:

1. **Appendicitis** dataset, a medical dataset suggested in [67].
2. **Australian** dataset [68], an economic dataset.
3. **Balance** dataset [69], used for the prediction of psychological states.
4. **Cleveland** dataset, related to heart diseases [70,71].
5. **Bands** dataset, a dataset related to printing problems [72].
6. **Dermatology** dataset [73], which is a medical dataset.
7. **Hayes-roth** dataset [74].
8. **Heart** dataset [75], a medical dataset.
9. **HouseVotes** dataset [76].
10. **Ionosphere** dataset, a dataset from the Johns Hopkins database [77,78].
11. **Liverdisorder** dataset [79], a medical dataset about liver disorders.
12. **Lymography** dataset [80].
13. **Mammographic** dataset [81], which is a dataset about breast cancer.
14. **Parkinsons** dataset, a medical dataset about Parkinson's Disease (PD) [82].
15. **Pima** dataset, a medical dataset [83].
16. **Popfailures** dataset [84], a dataset about climate.

17. **Spiral** dataset: The spiral artificial dataset contains 1000 two-dimensional examples that belong to two classes (500 examples each). The number of features is 2. The data in the first class were created using the following formula: $x_1 = 0.5t\cos(0.08t)$, $x_2 = 0.5t\cos\left(0.08t + \frac{\pi}{2}\right)$, and the second class data using: $x_1 = 0.5t\cos(0.08t + \pi)$, $x_2 = 0.5t\cos\left(0.08t + \frac{3\pi}{2}\right)$.
18. **Regions2** dataset, described in [85].
19. **Saheart** dataset [86], which is related to heart diseases.
20. **Segment** dataset [87], which is related to image processing.
21. **Wdbc** dataset [88], which is related to breast tumors.
22. **Wine** dataset. The wine recognition dataset contains data from wine chemical analysis. It contains 178 examples of 13 features each, which are classified into three classes. It has been examined in many published works [89,90].
23. **Eeg** dataset. As a real-word example, an EEG dataset described in [91] was used here. The datasets derived from the dataset are denoted as Z_F_S, ZONF_S, and ZO_NF_S.
24. **Zoo** dataset [92], used for the classification of animals.

The regression datasets were as follows:

1. **Abalone** dataset [93].
2. **Airfoil** dataset, a dataset from NASA related to aerodynamic and acoustic tests [94].
3. **Baseball** dataset, a dataset used to predict the points scored by baseball players.
4. **BK** dataset [95], used to estimate the points scored per minute in a basketball game.
5. **BL** dataset; this dataset is related to an experiment on the affects of machine adjustments on the time to count bolts.
6. **Concrete** dataset, related to civil engineering [96].
7. **Dee** dataset, used to predict the daily average price of electric energy in Spain.
8. **Diabetes** dataset, a medical dataset.
9. **FA** dataset, related to fat measurements.
10. **Housing** dataset, described in [97].
11. **MB** dataset, a statistics dataset [95].
12. **MORTGAGE** dataset, which contains economic data.
13. **NT** dataset, derived from [98].
14. **PY** dataset (the Pyrimidines problem) [99].
15. **Quake** dataset, which contains data from earthquakes [100].
16. **Treasure** dataset, which contains economic data.
17. **Wankara** dataset, which is about weather measurement

3.2. Experimental Results

The RBF network for the tests was coded in ANSI C++ with the help of the freely available Armadillo library [101]. In addition, in order to have greater reliability of the experimental results, a 10-fold validation technique was used. All the experiments were executed 30 times with different seeds for the random generator each time, and the average was measured. For the classification datasets, the average classification error is reported and, for the regression datasets, the mean test error. The machine used for the experiments was an AMD Ryzen 5950X with 128GB of RAM. The used operating system was Debian Linux. In order to accelerate the training process, the OpenMP library was incorporated [102]. The experimental settings are listed in Table 1. The experimental results for the classification datasets are listed in Table 2 and, for the regression datasets, in Table 3. For the experimental tables, the following were applied:

1. The column NN-PROP indicates the application of the Rprop method [103] in an artificial neural network [104,105] with 10 hidden nodes. The RPROP method is coded in the FCNN software package [106].
2. The column NN-GENETIC denotes the application of a genetic algorithm in the artificial neural network with 10 hidden nodes. The parameters of the used genetic algorithm are the same as in the second phase of the proposed method.

3. The column RBF-KMEANS denotes the classic training method for RBF networks by estimating centers and variances through K-means and the output weights by solving a linear system of equations.
4. The column IRBF-100 denotes the application of the current method with $\lambda = 100$.
5. The column IRBF-1000 denotes the application of the current method with $\lambda = 1000$.
6. In both tables, an extra line is added, in which the mean error for each method is shown. This row is denoted by the name AVERAGE. This line also shows the number of times the corresponding method achieved the best result. This number is shown in parentheses.

Table 1. The used values for the experimental parameters. The first column denotes the name of the parameter and the second the used value.

Parameter	**Value**
N_c	200
N_g	100
N_s	50
c_1	1.0
c_2	1.0
F	5.0
B	100.0
k	10
p_s	0.90
p_m	0.05

Table 2. Experimental results for the classification datasets. The first column is the name of the used dataset.

Dataset	**NN-RPROP**	**NN-GENETIC**	**RBF-KMEANS**	**IRBF-100**	**IRBF-1000**
Appendicitis	16.30%	18.10%	**12.23%**	16.47%	14.03%
Australian	36.12%	32.21%	34.89%	23.61%	**22.39%**
Balance	**8.81%**	8.97%	33.42%	12.65%	13.15%
Bands	36.32%	**35.75%**	37.22%	37.38%	36.29%
Cleveland	61.41%	51.60%	67.10%	49.77%	**49.64%**
Dermatology	**15.12%**	30.58%	62.34%	38.24%	35.64%
Hayes Roth	37.46%	56.18%	64.36%	**33.62%**	34.13%
Heart	30.51%	28.34%	31.20%	15.91%	**15.60%**
HouseVotes	6.04%	6.62%	6.13%	4.77%	**3.90%**
Ionosphere	13.65%	15.14%	16.22%	8.64%	**7.52%**
Liverdisorder	40.26%	31.11%	30.84%	27.36%	**25.63%**
Lymography	24.67%	23.26%	25.31%	**19.12%**	20.02%
Mammographic	18.46%	19.88%	21.38%	**17.17%**	17.30%
Parkinsons	22.28%	18.05%	17.41%	15.51%	**13.59%**
Pima	34.27%	32.19%	25.78%	23.61%	**23.23%**

Table 2. *Cont.*

Dataset	NN-RPROP	NN-GENETIC	RBF-KMEANS	IRBF-100	IRBF-1000
Popfailures	**4.81%**	5.94%	7.04%	5.21%	5.10%
Regions2	27.53%	29.39%	38.29%	26.08%	**25.77%**
Saheart	34.90%	34.86%	32.19%	**27.94%**	28.91%
Segment	52.14%	57.72%	59.68%	47.19%	**40.28%**
Spiral	46.59%	44.50%	44.87%	**19.43%**	19.56%
Wdbc	21.57%	8.56%	7.27%	**5.33%**	5.44%
Wine	30.73%	19.20%	31.41%	9.20%	**6.84%**
Z_F_S	29.28%	10.73%	13.16%	4.19%	**4.18%**
ZO_NF_S	6.43%	8.41%	9.02%	**4.31%**	4.35%
ZONF_S	27.27%	2.60%	4.03%	2.23%	**2.08%**
ZOO	15.47%	16.67%	21.93%	**10.13%**	11.13%
AVERAGE	**26.86%(3)**	**24.87%(1)**	**29.03%(1)**	**19.43%(8)**	**18.68%(13)**

Table 3. Experimental results for the regression datasets. The first column is the name of the used regression dataset.

DATASET	NN-RPROP	NN-GENETIC	RBF-KMEANS	IRBF-100	IRBF-1000
ABALONE	**4.55**	7.17	7.37	5.57	5.32
AIRFOIL	0.002	0.003	0.27	0.004	**0.003**
BASEBALL	92.05	103.60	93.02	**78.89**	85.58
BK	1.60	0.03	**0.02**	0.04	0.03
BL	4.38	5.74	0.013	**0.0003**	0.0003
CONCRETE	0.009	0.009	0.011	**0.007**	0.007
DEE	0.608	1.013	0.17	**0.16**	0.16
DIABETES	1.11	19.86	**0.49**	0.78	0.89
HOUSING	74.38	43.26	57.68	**20.27**	21.54
FA	0.14	1.95	**0.015**	0.032	0.029
MB	0.55	3.39	2.16	0.12	**0.09**
MORTGAGE	9.19	2.41	1.45	**0.39**	0.78
NT	0.04	**0.006**	8.14	0.007	0.007
PY	0.039	1.41	**0.012**	0.024	0.014
QUAKE	0.041	0.040	0.07	0.04	**0.03**
TREASURY	10.88	2.93	2.02	**0.33**	0.51
WANKARA	0.0003	0.012	**0.001**	0.002	0.002
AVERAGE	**11.71(1)**	**11.34(1)**	**10.17(5)**	**6.27(7)**	**6.76(3)**

As one can see from the experimental results, the proposed method significantly outperformed the other techniques in the majority of cases in terms of the average error in the test set. Moreover, the difference from the established method of training RBF networks was of the order of 40%, and in some cases, this percentage can be doubled. The statistical difference of the proposed technique against the rest is also shown in Figures 4 and 5. However, the proposed technique was significantly slower than the original training technique, as it is a two-stage technique. In the first stage, an optimal interval of values for the network parameters is created with a modified PSO method, and in the second stage, the network is trained using a genetic algorithm. Of course, this extra time can be significantly reduced by incorporating parallel techniques, as was done experimentally using the OpenMP library. Furthermore, changing the normalization factor λ from 100 to 1000 did not have much effect on the mean error in the test set. This implies that the proposed method is quite robust, since it does not have much dependence on this parameter.

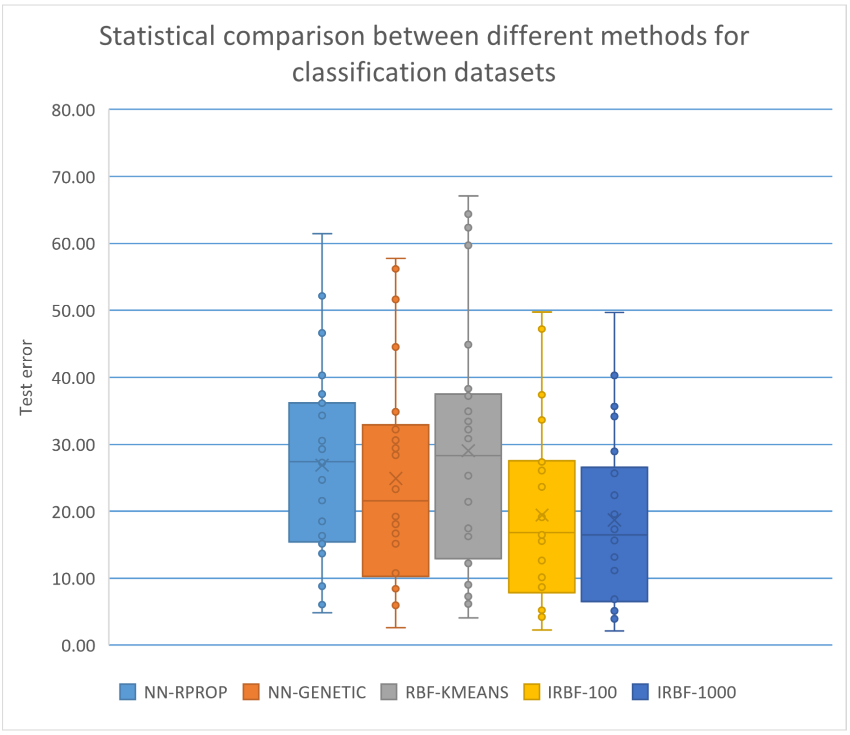

Figure 4. Graphical comparison of all methods for the classification datasets.

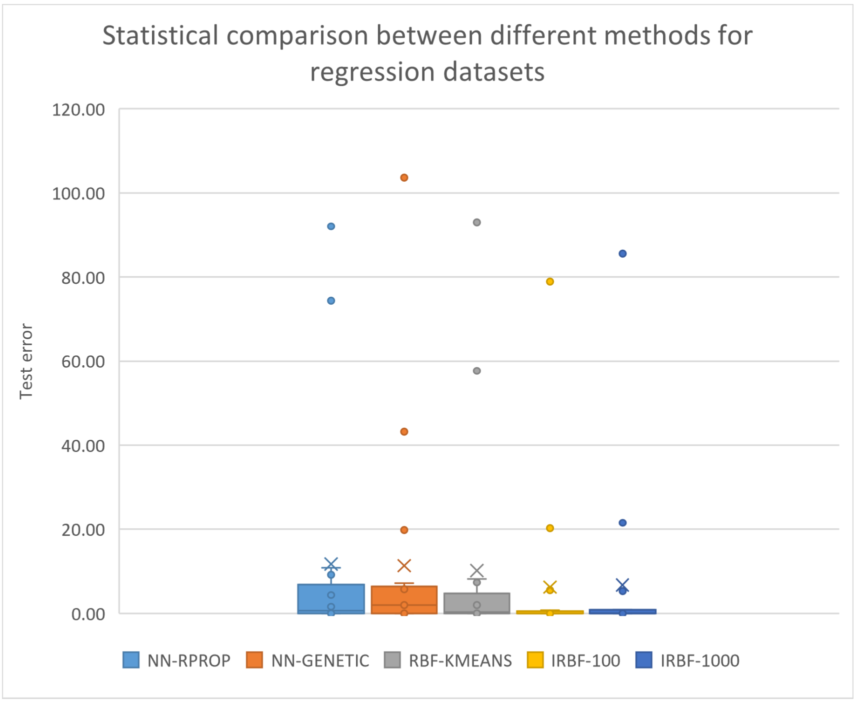

Figure 5. Graphical comparison of the methods for the regression datasets.

An additional experiment was performed with different values for the parameter F. The experimental results for this experiment are shown in Table 4 for the classification datasets and in Table 5 for the regression datasets. For this critical parameter, no large deviations appeared in the results of the proposed method. This further enhances the robustness and reliability of the proposed technique.

Furthermore, in the Table 6, the metrics of precision, recall, and f-score are shown for a series of classification datasets and for the proposed method (IRBF-100) and the classic method for training RBF networks (RBF-KMEANS). In these experimental results, the reader can see the superiority of the proposed technique over the traditional method of training RBF networks.

Table 4. Experimental results with the proposed method and using different values for the parameter F on the classification datasets.

DATASET	$F = 3$	$F = 5$	$F = 10$
Appendicitis	14.43%	14.03%	14.47%
Australian	23.45%	22.39%	23.21%
Balance	13.35%	13.15%	11.79%
Bands	36.48%	36.29%	36.76%
Cleveland	49.26%	49.64%	49.02%
Dermatology	36.54%	35.64%	34.37%
Hayes Roth	39.28%	34.13%	36.46%
Heart	15.14%	15.60%	14.89%
HouseVotes	4.93%	3.90%	6.41%
Ionosphere	7.56%	7.52%	9.05%
Liverdisorder	28.37%	25.63%	28.97%
Lymography	20.12%	20.02%	21.05%
Mammographic	18.04%	17.30%	18.21%
Parkinsons	18.51%	13.59%	13.49%
Pima	23.69%	23.23%	23.52%
Popfailures	5.76%	5.10%	4.50%
Regions2	25.79%	25.77%	25.32%
Saheart	28.89%	28.91%	26.99%
Segment	36.53%	40.28%	43.28%
Spiral	16.78%	19.56%	22.18%
Wdbc	4.64%	5.44%	5.10%
Wine	8.31%	6.84%	8.27%
Z_F_S	4.32%	4.18%	4.03%
ZO_NF_S	3.70%	4.35%	3.72%
ZONF_S	2.04%	2.08%	1.98%
ZOO	11.87%	11.13%	9.97%
AVERAGE	**18.65%**	**18.68%**	**19.12%**

Table 5. Experimental results with the proposed method using different values for the parameter F on the classification datasets.

DATASET	$F = 3$	$F = 5$	$F = 10$
ABALONE	5.56	5.32	5.41
AIRFOIL	0.004	0.003	0.004
BASEBALL	88.40	85.58	84.43
BK	0.03	0.03	0.02
BL	0.0005	0.0003	0.0002
CONCRETE	0.009	0.007	0.007
DEE	0.18	0.16	0.16
DIABETES	0.67	0.89	0.77
HOUSING	20.03	21.54	20.84
FA	0.03	0.029	0.036
MB	0.19	0.09	0.26
MORTGAGE	0.89	0.78	0.03
NT	0.006	0.007	0.007
PY	0.027	0.014	0.018
QUAKE	0.04	0.03	0.04
TREASURY	0.77	0.51	0.17
WANKARA	0.002	0.002	0.002
AVERAGE	**6.87**	**6.76**	**6.60**

Table 6. Precision, recall, and f-score for a series of classification datasets.

RBF-KMEANS	IRBF-100					
DATASET	PRECISION	RECALL	F-SCORE	PRECISION	RECALL	F-SCORE
APPENDICITIS	0.80	0.77	0.76	0.79	0.74	0.78
AUSTRALIAN	0.67	0.61	0.58	0.79	0.76	0.76
BALANCE	0.74	0.76	0.64	0.75	0.78	0.76
BANDS	0.52	0.51	0.48	0.58	0.57	0.56
HEART	0.68	0.69	0.67	0.86	0.85	0.85
IONOSPHERE	0.84	0.81	0.81	0.92	0.89	0.90
LIVERDISORDER	0.65	0.64	0.64	0.72	0.71	0.71
MAMMOGRAPHIC	0.81	0.81	0.81	0.83	0.83	0.82
PARKINSONS	0.76	0.68	0.69	0.85	0.80	0.81
PIMA	0.72	0.67	0.68	0.75	0.70	0.71
SAHEART	0.65	0.61	0.61	0.70	0.66	0.67
SEGMENT	0.43	0.39	0.39	0.58	0.53	0.53
SPIRAL	0.56	0.56	0.55	0.70	0.70	0.70
WDBC	0.93	0.91	0.92	0.96	0.94	0.95
WINE	0.74	0.65	0.66	0.93	0.93	0.92
Z_F_S	0.85	0.84	0.83	0.96	0.97	0.96
ZO_NF_S	0.90	0.90	0.90	0.95	0.95	0.95

4. Conclusions

In the present work, a two-stage hybrid method was proposed to efficiently identify the parameters of RBF neural networks. In the first stage of the method, a technique rooted in particle swarm optimization was used to efficiently identify a reliable interval of values for the neural network parameters. In the second stage of the method, an intelligent global optimization technique was used to locate the neural network parameters within the optimal value interval of the first stage. In this work, a genetic algorithm was used in the second phase, but any global optimization method could be used in its place.

The method was applied to a multitude of classification and regression problems from the relevant literature. In almost all cases, the proposed method significantly outperformed the other machine learning models, and on average, the improvement in the error on the test sets was of the order of 40% relative to the established RBF training method. Moreover, the method is quite robust with respect to the basic parameters since any changes in the parameter values do not significantly affect its performance. Furthermore, the method can efficiently locate the value interval of network parameters without any prior knowledge about the type of training data or whether it is a classification or a regression problem. However, the proposed technique is significantly more time-consuming than the traditional training technique, as it requires computational time for both of its phases. However, this effect can be overcome to some extent by the use of modern parallel computing techniques.

The method could be extended by the use of other techniques of training the parameters in RBF networks, such as, for example, the differential evolutionary method [107]. Furthermore, more efficient methods of terminating the first stage of the method could be used, as finding a suitable interval of values for the network parameters requires many numerical calculations.

Author Contributions: I.G.T. and V.C. conceived of the idea and methodology and supervised the technical part regarding the software. I.G.T. conducted the experiments, employing datasets, and provided the comparative experiments. V.C. performed the statistical analysis and prepared the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The experiments of this research work were performed at the high-performance computing system established at the Knowledge and Intelligent Computing Laboratory, Department of Informatics and Telecommunications, University of Ioannina, acquired with the project "Educational Laboratory equipment of TEI of Epirus" with MIS 5007094 funded by the Operational Programme "Epirus" 2014–2020, by ERDF and national funds.

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

References

1. Mjahed, M. The use of clustering techniques for the classification of high energy physics data. *Nucl. Instrum. Methods Phys. Res. Sect. A* **2006**, *559*, 199–202. [CrossRef]
2. Andrews, M.; Paulini, M.; Gleyzer, S.; Poczos, B. End-to-End Event Classification of High-Energy Physics Data. *J. Phys.* **2018**, *1085*, 42022. [CrossRef]
3. He, P.; Xu, C.J.; Liang, Y.Z.; Fang, K.T. Improving the classification accuracy in chemistry via boosting technique. *Chemom. Intell. Lab. Syst.* **2004**, *70*, 39–46. [CrossRef]
4. Aguiar, J.A.; Gong, M.L.; Tasdizen, T. Crystallographic prediction from diffraction and chemistry data for higher throughput classification using machine learning. *Comput. Mater. Sci.* **2020**, *173*, 109409. [CrossRef]
5. Kaastra, I.; Boyd, M. Designing a neural network for forecasting financial and economic time series. *Neurocomputing* **1996**, *10*, 215–236. [CrossRef]
6. Hafezi, R.; Shahrabi, J.E. Hadavandi, A bat-neural network multi-agent system (BNNMAS) for stock price prediction: Case study of DAX stock price. *Appl. Soft Comput.* **2015**, *29*, 196–210. [CrossRef]

7. Yadav, S.S.; Jadhav, S.M. Deep convolutional neural network based medical image classification for disease diagnosis. *J. Big Data* **2019**, *6*, 113. [CrossRef]
8. Qing, L.; Linhong, W.; Xuehai, D. A Novel Neural Network-Based Method for Medical Text Classification. *Future Internet* **2019**, *11*, 255. [CrossRef]
9. Park, J.; Sandberg, I.W. Universal Approximation Using Radial-Basis-Function Networks. *Neural Comput.* **1991**, *3*, 246–257. [CrossRef]
10. Ghosh, J.; Nag, A. An Overview of Radial Basis Function Networks. In *Radial Basis Function Networks 2. Studies in Fuzziness and Soft Computing*; Howlett, R.J., Jain, L.C., Eds.; Physica: Heidelberg, Germany, 2001; Volume 67.
11. Nam, M.-D.; Thanh, T.-C. Numerical solution of differential equations using multiquadric radial basis function networks. *Neural Netw.* **2001**, *14*, 185–199.
12. Mai-Duy, N. Solving high order ordinary differential equations with radial basis function networks. *Int. J. Numer. Meth. Eng.* **2005**, *62*, 824–852. [CrossRef]
13. Laoudias, C.; Kemppi, P.; Panayiotou, C.G. Localization Using Radial Basis Function Networks and Signal Strength Fingerprints. In Proceedings of the WLAN, GLOBECOM 2009—2009 IEEE Global Telecommunications Conference, Honolulu, HI, USA, 30 November–4 December 2009; pp. 1–6.
14. Azarbad, M.; Hakimi, S.; Ebrahimzadeh, A. Automatic recognition of digital communication signal. *Int. J. Energy* **2012**, *3*, 21–33.
15. Teng, P. Machine-learning quantum mechanics: Solving quantum mechanics problems using radial basis function networks. *Phys. Rev. E* **2018**, *98*, 33305. [CrossRef]
16. Jovanović, R.; Sretenovic, A. Ensemble of radial basis neural networks with K-means clustering for heating energy consumption prediction. *Fme Trans.* **2017**, *45*, 51–57. [CrossRef]
17. Yu, D.L.; Gomm, J.B.; Williams, D. Sensor fault diagnosis in a chemical process via RBF neural networks. *Control. Eng. Pract.* **1999**, *7*, 49–55. [CrossRef]
18. Shankar, V.; Wright, G.B.; Fogelson, A.L.; Kirby, R.M. A radial basis function (RBF) finite difference method for the simulation of reaction–diffusion equations on stationary platelets within the augmented forcing method. *Int. J. Numer. Meth. Fluids* **2014**, *75*, 1–22. [CrossRef]
19. Shen, W.; Guo, X.; Wu, C.; Wu, D. Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm. *Knowl.-Based Syst.* **2011**, *24*, 378–385. [CrossRef]
20. Momoh, J.A.; Reddy, S.S. Combined Economic and Emission Dispatch using Radial Basis Function. In Proceedings of the 2014 IEEE PES General Meeting Conference & Exposition, National Harbor, MD, USA, 27–31 July 2014; pp. 1–5.
21. Sohrabi, P.; Shokri, B.J.; Dehghani, H. Predicting coal price using time series methods and combination of radial basis function (RBF) neural network with time series. *Miner. Econ.* **2021**, 1–10. [CrossRef]
22. Ravale, U.; Marathe, N.; Padiya, P. Feature Selection Based Hybrid Anomaly Intrusion Detection System Using K Means and RBF Kernel Function. *Procedia Comput. Sci.* **2015**, *45*, 428–435. [CrossRef]
23. Lopez-Martin, M.; Sanchez-Esguevillas, A.; Arribas, J.I.; Carro, B. Network Intrusion Detection Based on Extended RBF Neural Network With Offline Reinforcement Learning. *IEEE Access* **2021**, *9*, 153153–153170. [CrossRef]
24. Yu, H.T.; Xie, T.; Paszczynski, S.; Wilamowski, B.M. Advantages of Radial Basis Function Networks for Dynamic System Design. *IEEE Trans. Ind. Electron.* **2011**, *58*, 5438–5450. [CrossRef]
25. Yokota, R.; Barba, L.A.; Knepley, M.G. PetRBF—A parallel O(N) algorithm for radial basis function interpolation with Gaussians. *Comput. Methods Appl. Mech. Eng.* **2010**, *199*, 1793–1804. [CrossRef]
26. Lu, C.; Ma, N.; Wang, Z. Fault detection for hydraulic pump based on chaotic parallel RBF network. *EURASIP J. Adv. Signal Process.* **2011**, *2011*, 49. [CrossRef]
27. Kuncheva, L.I. Initializing of an RBF network by a genetic algorithm. *Neurocomputing* **1997**, *14*, 273–288. [CrossRef]
28. Ros, F.; Pintore, M.; Deman, A.; Chrétien, J.R. Automatical initialization of RBF neural networks. *Chemom. Intell. Lab. Syst.* **2007**, *87*, 26–32. [CrossRef]
29. Wang, D.; Zeng, X.J.; Keane, J.A. A clustering algorithm for radial basis function neural network initialization. *Neurocomputing* **2012**, *77*, 144–155. [CrossRef]
30. Ricci, E.; Perfetti, R. Improved pruning strategy for radial basis function networks with dynamic decay adjustment. *Neurocomputing* **2006**, *69*, 1728–1732. [CrossRef]
31. Huang, G.-B.P. Saratchandran and N. Sundararajan, A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation. *IEEE Trans. Neural Netw.* **2005**, *16*, 57–67. [CrossRef] [PubMed]
32. Bortman, M.; Aladjem, M. A Growing and Pruning Method for Radial Basis Function Networks. *IEEE Trans. Neural. Netw.* **2009**, *20*, 1039–1045. [CrossRef] [PubMed]
33. Karayiannis, N.B.; Randolph-Gips, M.M. On the construction and training of reformulated radial basis function neural networks. *IEEE Trans. Neural Netw.* **2003**, *14*, 835–846. [CrossRef]
34. Peng, J.X.; Li, K.; Huang, D.S. A Hybrid Forward Algorithm for RBF Neural Network Construction. *IEEE Trans. Neural Netw.* **2006**, *17*, 1439–1451. [CrossRef]
35. Du, D.; Li, K.; Fei, M. A fast multi-output RBF neural network construction method. *Neurocomputing* **2010**, *73*, 2196–2202. [CrossRef]
36. Marini, F.; Walczak, B. Particle swarm optimization (PSO). A tutorial. *Chemom. Intell. Lab. Syst.* **2015**, *149*, 153–165. [CrossRef]

37. Liu, B.; Wang, L.; Jin, Y.H. An Effective PSO-Based Memetic Algorithm for Flow Shop Scheduling. *IEEE Trans. Syst. Cybern. Part B* **2007**, *37*, 18–27. [CrossRef] [PubMed]
38. Yang, J.; He, L.; Fu, S. An improved PSO-based charging strategy of electric vehicles in electrical distribution grid. *Appl. Energy* **2014**, *128*, 82–92. [CrossRef]
39. Mistry, K.; Zhang, L.; Neoh, S.C.; Lim, C.P.; Fielding, B. A Micro-GA Embedded PSO Feature Selection Approach to Intelligent Facial Emotion Recognition. *IEEE Trans. Cybern.* **2017**, *47*, 1496–1509. [CrossRef] [PubMed]
40. Han, S.; Shan, X.; Fu, J.; Xu, W.; Mi, H. Industrial robot trajectory planning based on improved pso algorithm. *J. Phys. Conf. Ser.* **2021**, *1820*, 12185. [CrossRef]
41. Floudas, C.A.; Gounaris, C.E. A review of recent advances in global optimization. *J. Glob. Optim.* **2009**, *45*, 3–38. [CrossRef]
42. Goldberg, D. Genetic Algorithms. In *Search, Optimization and Machine Learning*; Addison-Wesley Publishing Company: Reading, MA, USA, 1989.
43. Michaelewicz, Z. *Genetic Algorithms + Data Structures = Evolution Programs*; Springer: Berlin, Germany, 1996.
44. Grady, S.A.; Hussaini, M.Y.; Abdullah, M.M. Placement of wind turbines using genetic algorithms. *Renew. Energy* **2005**, *30*, 259–270. [CrossRef]
45. Agarwal, V.; Bhanot, S. Radial basis function neural network-based face recognition using firefly algorithm. *Neural. Comput. Appl.* **2018**, *30*, 2643–2660. [CrossRef]
46. Jiang, S.; Lu, C.; Zhang, S.; Lu, X.; Tsai, S.-B.; Wang, C.-K.; Gao, Y.; Shi, Y.; Lee, C.-H. Prediction of Ecological Pressure on Resource-Based Cities Based on an RBF Neural Network Optimized by an Improved ABC Algorithm. *IEEE Access* **2019**, *7*, 47423–47436. [CrossRef]
47. Wang, H.; Wang, W.; Zhou, X.; Sun, H.; Zhao, J.; Yu, X.; Cui, Z. Firefly algorithm with neighborhood attraction, Information. *Sciences* **2017**, *382–383*, 374–387.
48. Khan, I.U.; Aslam, N.; Alshehri, R.; Alzahrani, S.; Alghamdi, M.; Almalki, A.; Balabeed, M. Cervical Cancer Diagnosis Model Using Extreme Gradient Boosting and Bioinspired Firefly Optimization. *Sci. Program.* **2021**, *2021*, 5540024. [CrossRef]
49. Zivkovic, M.; Bacanin, N.; Antonijevic, M.; Nikolic, B.; Kvascev, G.; Marjanovic, M.; Savanovic, N. Hybrid CNN and XGBoost Model Tuned by Modified Arithmetic Optimization Algorithm for COVID-19 Early Diagnostics from X-ray Images. *Electronics* **2022**, *11*, 3798. [CrossRef]
50. MacQueen, J. Some methods for classification and analysis of multivariate observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, CA, USA, 21 June–18 July 1965; Volume 1, pp. 281–297.
51. Hansen, E.; Walster, G.W. *Global Optimization Using Interval Analysis*; Marcel Dekker Inc.: New York, NY, USA, 2004.
52. Markót, M.; Fernández, J.; Casado, L.G.; Csendes, T. New interval methods for constrained global optimization. *Math. Program.* **2006**, *106*, 287–318. [CrossRef]
53. Žilinskas, A.; Žilinskas, J. Interval Arithmetic Based Optimization in Nonlinear Regression. *Informatica* **2010**, *21*, 149–158. [CrossRef]
54. Schnepper, C.A.; Stadtherr, M.A. Robust process simulation using interval methods. *Comput. Chem. Eng.* **1996**, *20*, 187–199. [CrossRef]
55. Carreras, C.; Walker, I.D. Interval methods for fault-tree analysis in robotics. *IEEE Trans. Reliab.* **2001**, *50*, 3–11. [CrossRef]
56. Serguieva, A.; Hunte, J. Fuzzy interval methods in investment risk appraisal. *Fuzzy Sets Syst.* **2004**, *142*, 443–466. [CrossRef]
57. Poli, R.; kennedy, J.K.; Blackwell, T. Particle swarm optimization An Overview. *Swarm Intell.* **2007**, *1*, 33–57. [CrossRef]
58. Trelea, I.C. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Inf. Process. Lett.* **2003**, *85*, 317–325. [CrossRef]
59. Shi, Y.; Eberhart, R.C. Empirical study of particle swarm optimization. In Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Washington, DC, USA, 6–9 July 1999; Volume 3, pp. 1945–1950.
60. Borowska, B. Exponential Inertia Weight in Particle Swarm Optimization. In *Information Systems Architecture and Technology: Proceedings of 37th International Conference on Information Systems Architecture and Technology—ISAT 2016—Part IV*; Wilimowska, Z., Borzemski, L., Grzech, A., Świątek, J., Eds.; Springer: Cham, Switzerland, 2017; Volume 524.
61. Zhang, L.; Yu, H.; Hu, S. A New Approach to Improve Particle Swarm Optimization. In *Genetic and Evolutionary Computation—GECCO 2003*; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2723.
62. Borowska, B. Dynamic Inertia Weight in Particle Swarm Optimization. In *Advances in Intelligent Systems and Computing II. CSIT 2017*; Shakhovska, N., Stepashko, V., Eds.; Springer: Cham, Switzerland, 2018; Volume 689.
63. Shi, Y.; Eberhart, R.C. Fuzzy adaptive particle swarm optimization. In Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546), Seoul, Republic of Korea, 27–30 May 2001; Volume 1, pp. 101–106.
64. Kaelo, P.; Ali, M.M. Integrated crossover rules in real coded genetic algorithms. *Eur. J. Oper. Res.* **2007**, *176*, 60–76. [CrossRef]
65. Tsoulos, I.G. Modifications of real code genetic algorithm for global optimization. *Appl. Math. Comput.* **2008**, *203*, 598–607. [CrossRef]
66. Alcalá-Fdez, J.; Fernandez, A.; Luengo, J.; Derrac, J.; García, S.; Sánchez, L.; Herrera, F. KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. *J. Mult. Valued Log. Soft Comput.* **2011**, *17*, 255–287.
67. Weiss, S.M.; Kulikowski, C.A. *Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning and Expert Systems*; Morgan Kaufmann Publishers Inc.; Morgan Kaufmann Publishing: San Mateo CA, USA, 1991.
68. Quinlan, J.R. Simplifying Decision Trees. *Int. -Man-Mach. Stud.* **1987**, *27*, 221–234. [CrossRef]

69. Shultz, T.; Mareschal, D.; Schmidt, W. Modeling Cognitive Development on Balance Scale Phenomena. *Mach. Learn.* **1994**, *16*, 59–88. [CrossRef]
70. Zhou, Z.H.; Jiang, Y. NeC4.5: Neural ensemble based C4.5. *IEEE Trans. Knowl. Data Eng.* **2004**, *16*, 770–773. [CrossRef]
71. Setiono, R.; Leow, W.K. FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks. *Appl. Intell.* **2000**, *12*, 15–25. [CrossRef]
72. Evans, B.; Fisher, D. Overcoming process delays with decision tree induction. *IEEE Expert.* **1994**, *9*, 60–66. [CrossRef]
73. Demiroz, G.; Govenir, H.A.; Ilter, N. Learning Differential Diagnosis of Eryhemato-Squamous Diseases using Voting Feature Intervals. *Artif. Intell. Med.* **1998**, *13*, 147–165.
74. Hayes-Roth, B.; Hayes-Roth, B.F. Concept learning and the recognition and classification of exemplars. *J. Verbal Learning Verbal Behav.* **1977**, *16*, 321–338. [CrossRef]
75. Kononenko, I.; Šimec, E.; Robnik-Šikonja, M. Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF. *Appl. Intell.* **1997**, *7*, 39–55. [CrossRef]
76. French, R.M.; Chater, N. Using noise to compute error surfaces in connectionist networks: A novel means of reducing catastrophic forgetting. *Neural Comput.* **2002**, *14*, 1755–1769. [CrossRef] [PubMed]
77. Dy, J.G.; Brodley, C.E. Feature Selection for Unsupervised Learning. *J. Mach. Learn. Res.* **2004**, *5*, 845–889.
78. Perantonis, S.J.; Virvilis, V. Input Feature Extraction for Multilayered Perceptrons Using Supervised Principal Component Analysis. *Neural Process. Lett.* **1999**, *10*, 243–252. [CrossRef]
79. Garcke, J.; Griebel, M. Classification with sparse grids using simplicial basis functions. *Intell. Data Anal.* **2002**, *6*, 483–502. [CrossRef]
80. Cestnik, G.; Konenenko, I.; Bratko, I. Assistant-86: A Knowledge-Elicitation Tool for Sophisticated Users. In *Progress in Machine Learning*; Bratko, I., Lavrac, N., Eds.; Sigma Press: Wilmslow, UK, 1987; pp. 31–45.
81. Elter, M.R.; Schulz-Wendtland, T.W. The prediction of breast cancer biopsy outcomes using two CAD approaches that both emphasize an intelligible decision process. *Med. Phys.* **2007**, *34*, 4164–4172. [CrossRef]
82. Little, M.A.; McSharry, P.E.; Hunter, E.J.; Spielman, J.; Ramig, L.O. Suitability of dysphonia measurements for telemonitoring of Parkinson's disease. *IEEE Trans. Biomed. Eng.* **2009**, *56*, 1015. [CrossRef] [PubMed]
83. Smith, J.W.; Everhart, J.E.; Dickson, W.C.; Knowler, W.C.; Johannes, R.S. Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In Proceedings of the Symposium on Computer Applications and Medical Care IEEE Computer Society Press in Medical Care, Orlando, FL, USA, 7–11 November 1988; pp. 261–265.
84. Lucas, D.D.; Klein, R.; Tannahill, J.; Ivanova, D.; Brandon, S.; Domyancic, D.; Zhang, Y. Failure analysis of parameter-induced simulation crashes in climate models. *Geosci. Model Dev.* **2013**, *6*, 1157–1171. [CrossRef]
85. Giannakeas, N.; Tsipouras, M.G.; Tzallas, A.T.; Kyriakidi, K.; Tsianou, Z.E.; Manousou, P.; Hall, A.; Karvounis, E.C.; Tsianos, V.; Tsianos, E. A clustering based method for collagen proportional area extraction in liver biopsy images. In Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, New Orleans, LA, USA, 4–7 November 1988; pp. 3097–3100.
86. Hastie, T.; Tibshirani, R. Non-parametric logistic and proportional odds regression. *JRSS-C* **1987**, *36*, 260–276. [CrossRef]
87. Dash, M.; Liu, H.; Scheuermann, P.; Tan, K.L. Fast hierarchical clustering and its validation. *Data Knowl. Eng.* **2003**, *44*, 109–138. [CrossRef]
88. Wolberg, W.H.; Mangasarian, O.L. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proc. Natl. Acad. Sci. USA* **1990**, *87*, 9193–9196. [CrossRef]
89. Raymer, M.; Doom, T.E.; Kuhn, L.A.; Punch, W.F. Knowledge discovery in medical and biological datasets using a hybrid Bayes classifier/evolutionary algorithm. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2003**, *33*, 802–813. [CrossRef]
90. Zhong, P.; Fukushima, M. Regularized nonsmooth Newton method for multi-class support vector machines. *Optim. Methods Softw.* **2007**, *22*, 225–236. [CrossRef]
91. Andrzejak, R.G.; Lehnertz, K.; Mormann, F.; Rieke, C.; David, P.; Elger, C.E. Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state. *Phys. Rev. E* **2001**, *64*, 1–8. [CrossRef]
92. Koivisto, M.; Sood, K. Exact Bayesian Structure Discovery in Bayesian Networks. *J. Mach. Learn. Res.* **2004**, *5*, 549–573.
93. Nash, W.J.; Sellers, T.L.; Talbot, S.R.; Cawthor, A.J.; Ford, W.B. *The Population Biology of Abalone (Haliotis Species) in Tasmania. I. Blacklip Abalone (H. rubra) from the North Coast and Islands of Bass Strait, Sea Fisheries Division*; Technical Report No. 48; Department of Primary Industry and Fisheries, Tasmania: Hobart, Australia, 1994.
94. Brooks, T.F.; Pope, D.S.; Marcolini, A.M. *Airfoil Self-Noise and Prediction*; Technical Report, NASA RP-1218; National Aeronautics and Space Administration: Washington, DC, USA, 1989.
95. Simonoff, J.S. *Smooting Methods in Statistics*; Springer: Berlin/Heidelberg, Germany, 1996.
96. Cheng, Y.I. Modeling of strength of high performance concrete using artificial neural networks. *Cem. Concr. Res.* **1998**, *28*, 1797–1808.
97. Harrison, D.; Rubinfeld, D.L. Hedonic prices and the demand for clean ai. *J. Environ. Econ. Manag.* **1978**, *5*, 81–102. [CrossRef]
98. Mackowiak, P.A.; Wasserman, S.S.; Levine, M.M. A critical appraisal of 98.6 degrees f, the upper limit of the normal body temperature, and other legacies of Carl Reinhold August Wunderlich. *J. Amer. Med. Assoc.* **1992**, *268*, 1578–1580. [CrossRef]

99. King, R.D.; Muggleton, S.; Lewis, R.; Sternberg, M.J.E. Drug design by machine learning: The use of inductive logic programming to model the structure-activity relationships of trimethoprim analogues binding to dihydrofolate reductase. *Proc. Nat. Acad. Sci. USA* **1992**, *89*, 11322–11326. [CrossRef] [PubMed]
100. Sikora, M.; Wrobel, L. Application of rule induction algorithms for analysis of data collected by seismic hazard monitoring systems in coal mines. *Arch. Min. Sci.* **2010**, *55*, 91–114.
101. Sanderson, C.; Curtin, R. Armadillo: A template-based C++ library for linear algebra. *J. Open Source Softw.* **2016**, *1*, 26. [CrossRef]
102. Dagum, L.; Menon, R. OpenMP: An industry standard API for shared-memory programming. *IEEE Comput. Sci. Eng.* **1998**, *5*, 46–55. [CrossRef]
103. Riedmiller, M.; Braun, H. A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP algorithm. In Proceedings of the IEEE International Conference on Neural Networks, San Francisco, CA, USA, 28 March–1 April 1993; pp. 586–591.
104. Bishop, C. *Neural Networks for Pattern Recognition*; Oxford University Press: Oxford, UK, 1995.
105. Cybenko, G. Approximation by superpositions of a sigmoidal function. *Math. Control. Signals Syst.* **1989**, *2*, 303–314. [CrossRef]
106. Klima, G. Fast Compressed Neural Networks. Available online: http://fcnn.sourceforge.net/ (accessed on 5 January 2023).
107. Das, S.; Suganthan, P.N. Differential Evolution: A Survey of the State-of-the-Art. *IEEE Trans. Evol.* **2011**, *15*, 4–31. [CrossRef]