Article

Locate the parameters of RBF networks using a hybrid Particle Swarm Optimization method

Ioannis G. Tsoulos[1,†,‡,*], **Vasileios Charilogis**[2,†] [2,]

[1] Department of Informatics and Telecommunications, University of Ioannina, Greece; itsoulos@uoi.gr
[2] Department of Informatics and Telecommunications, University of Ioannina, Greece; v.charilog@uoi.gr
* Correspondence: itsoulos@uoi.gr;
† Current address: Department of Informatics and Telecommunications, University of Ioannina, Greece.
‡ 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 find 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 is 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 well-known 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. Locate the parameters of RBF networks using a hybrid Particle Swarm Optimization method. *Journal Not Specified* **2022**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

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 areas 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 expressed as a 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 $\overrightarrow{x}$ is the input pattern to the equation. The number of values 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 $y(\overrightarrow{x})$ is 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 defined as:

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

A plot for the Gaussian function with $c = 0$, $\sigma = 1$ is displayed in Figure 1. As it is observed, the value of the function decreases as we move away from the center. An extensive overview of RBF networks is given in the work of Ghosh and Nag [10]. RBF networks are used as approximation tools in various cases, such as solutions of differential equations [11,12], digital communications [13,14], physics [15,16], chemistry [17,18], economics [19–21], network security [22,23] etc. RBF networks are thoroughly discussed in [24] and it has 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], construction of RBF networks [33–35] etc.

In the current 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 is collected from the training data of the neural network and an attempt is made to identify a small interval of values for the neural network parameters. To identify this interval, an optimization method is used, which gradually creates the optimal value interval, which is estimated to give the lowest value for the training error of the network. To locate the optimal interval, the particle swarm optimization (PSO) technique is used [38]. 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 [39], developing charging strategies for electric vehicles [40], emotion recognition [41], robot trajectory planning [42] etc. The detection of the value interval is performed in order to then make the minimization of the network error faster and more efficient in the second phase optimization method. In the second phase, the parameters of the neural network are optimized within the optimal value interval of the first phase. The optimization can be performed by any global optimization method [43]. In this work, the genetic algorithms [45–47] were chosen for the second phase. The main advantages of genetic algorithms are tolerance on errors, easy to implement in parallel, efficient exploration of the search space etc.

Recently, many work has been appeared to tune the parameters of machine learning models, such as the work of Agarwal and Bhanot [36] for the adaptation of the RBF parameters, the incorporation of an improved ABC algorithm to adapt the parameters of RBF networks [37], the usage of the Firefly algorithm for optimization [44] along with machine learning models for Cervical cancer diagnosis [48], adaptation of 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 and finally in section 4 some conclusions are presented.

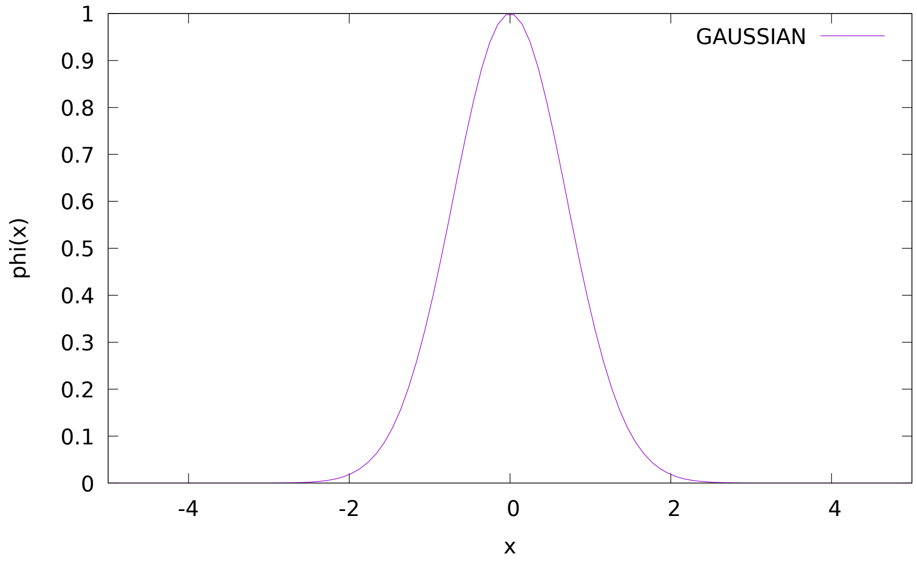

Figure 1. A typical plot for the Gaussian function, for $c = 0$ and $\sigma = 1$.

2. Method description

The training error of the RBF network is expressed as:

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

The value m stands for the number of patterns and t_i is the actual output for pattern x_i. The vector g is the set of the 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 estimated through 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 solution:

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

The proposed work uses two computational phases to optimally calculate the network parameters. In the first phase, a promising range for the parameters of the network is calculated through an optimization process that incorporates interval arithmetic. In the second phase, the parameters of the network are 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.

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 3.

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 1 The K-Means algorithm.

```
1. Repeat
   (a) $S_j = \{\},\ j = 1..k$
   (b) For each sample $x_i,\ i = 1,...,m$ do
       i.  Calculate $j^* = \min_{i=1}^{k}\left\{D\left(x_i, c_j\right)\right\}$.
       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. Calculate the variances for every center as
   $$\sigma_j^2 = \frac{\sum_{i=1}^{M_j}\left(x_i - c_j\right)^2}{M_j}$$
3. Terminate if there is no change in centers $c_j$.
```

2.2. The proposed PSO algorithm

During this phase, arithmetic interval techniques are used to find a suitable range for the parameters of the RBF network. The interval techniques [51–53] are 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 neural network. The initial bounding box is defined as S which is a subset of $\mathbb{R}^{\mathrm{n}}$:

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

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 the parameters of the network, a modified PSO algorithm [57] is used. The proposed variant of the PSO method is based on the original technique (algorithm 1 of[57]) but 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 a global optimization method 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 the algorithm 2. 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 the 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 the number of particles.
2. **Set** the normalization factor λ.
3. **Set** the k weights of the RBF network.

4. **Set** N_g the maximum number of 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_g particles. Each particle $p_i,\ i = 1, ..., N_c$ is considered as a set of intervals randomly initialized in S. The layout of each particle is graphically presented in Figure 2.
9. **For** $i = 1, ..., N_c$ **do**

    (a) **Calculate** the fitness f_i of particle p_i using the procedure outlined in Algorithm 3.

    (b) **If** $L^*(f_i, f^*) = \text{TRUE}$ **then** $f^* = f_i,\ p^* = p_i$

    (c) **Set** $p_{b,i} = p_i,\ f_{b,i} = f_i$ the best located position for particle i and the associated fitness value.

    (d) **For** $j = 1, ..., n$ **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 being 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 far as possible, excessive values for the velocity. This would result in the particles moving out of their value range very quickly and 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 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 and for low values it allows a more focused search.
13. **For** $i = 1, ..., N_c$ **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 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** $L^*(f_i, f_{b,i}) = \text{TRUE}$ **then** $p_{b,i} = p_i,\ f_{b,i} = f_i$

    (f) **If** $L^*(f_i, f^*) = \text{TRUE}$ **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 2 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
```

Figure 2. The layout of the particles in the proposed PSO algorithm.

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	...	w_k

Algorithm 3 Fitness calculation for the modified PSO algorithm.

The fitness calculation for a given particle g has 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 $f_g = [E_{min}(g), E_{max}(g)]$.
```

2.3. Optimization of parameters through genetic algorithm

During the second phase of the proposed method, a genetic algorithm is performed, 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 is enhanced using the stopping rule suggested by Tsoulos [65]. This genetic algorithm has the following steps:

1. **Initialization Step**
   (a) **Set** as N_c the number of chromosomes. Every chromosome is coded as in case of PSO using the scheme of Figure 2.
   (b) **Set** as N_g the maximum number of generations allowed.
   (c) **Set** k the number of nodes for the RBF network.
   (d) **Obtain** the domain range S from the procedure of subsection 2.2.
   (e) **Initialize** the N_C randomly in S.
   (f) **Set** the selection rate $p_s \in [0, 1]$
   (g) **Set** 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**
   Apply the genetic operations of selection, crossover and mutation.

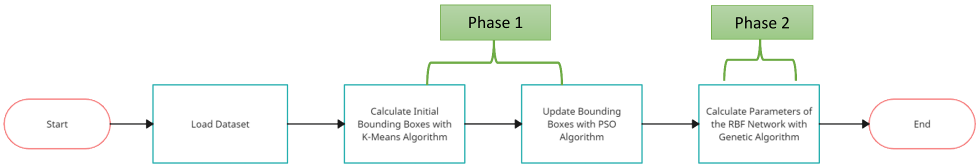
Phase 1
Phase 2
Start
Load Dataset
Calculate Initial Bounding Boxes with K-Means Algorithm
Update Bounding Boxes with PSO Algorithm
Calculate Parameters of the RBF Network with Genetic Algorithm
End

Figure 3. Graphical representation of the proposed method of two phases.

(a) **Selection procedure.** First, the population of chromosomes is sorted based on the associated fitness values. The best $(1 - p_s) \times N_c$ chromosomes are transferred unchanged to the next generation, while the remaining ones are replaced by offsprings created by the crossover procedure. During the selection step, a series of mating pairs are chosen using the well - known procedure of tournament selection for each parent.

(b) **Crossover procedure**: For each pair (z, w) of selected parents two new offsprings $\tilde{z}$ and $\tilde{w}$ are created with the following procedure:

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

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 are hold then **Terminate else Goto** Evaluation Step.

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

3. Experiments

The suggested method was tested on a series of classification and regression problems found in various papers and sites from 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 datasets was found in the Statlib URL ftp://lib.stat.cmu.edu/datasets/index.html(accessed on 5 January 2023).

3.1. Experimental datasets

The classification datasets that were used are the following:

1. **Appendictis** dataset, a medical dataset suggested in [67].
2. **Australian** dataset [68], an economic dataset.
3. **Balance** dataset [69], used for 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. This dataset[74] contains **5** numeric-valued attributes and 132 patterns.
8. **Heart** dataset [75], a medical dataset about heart diseases.
9. **HouseVotes** dataset [76], which is about votes in the U.S. House of Representatives Congressmen.
10. **Ionosphere** dataset a dataset found the Johns Hopkins database [77,78].
11. **Liverdisorder** dataset [79], a medical dataset about liver disorders.

12. **Lymography** dataset [80]. The aim here is to detect the presence of a lymphoma in patients.
13. **Mammographic** dataset [81], which is a dataset about breast cancer.
14. **Parkinsons** dataset, a medical dataset about the 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 the features is 2. The data in the first class are 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 that are classified into three classes. It has been examined in many published works [89,90].
23. **Eeg** dataset. As an real word example, consider an EEG dataset described in [91] is 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 classification of animals.

The regression datasets are:

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] and is 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 the electricity 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 [98].
12. **MORTGAGE** dataset, which contains economic data.
13. **NT** dataset, derived from [99].
14. **PY** dataset (Pyrimidines problem)[100].
15. **Quake** dataset, which contains data from earthquakes [101].
16. **Treasure** dataset, which contains economic data.
17. **Wankara** dataset, that is about weather measurement

3.2. Experimental results

The RBF network for the experiments was coded in ANSI C++ with the help of the freely available Armadillo library [102]. In addition, in order to have greater reliability in 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 was 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 [103]. The experimental settings are listed in the 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 are applied:

1. The column NN-PROP indicates the application of the Rprop method [104] in an artificial neural network [105,106] with 10 hidden nodes. The RPROP method is coded in the FCNN software package [107].
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 has been 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.

As one can see from the experimental results, the proposed method significantly outperforms 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 is 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 graphs 4 and 5. However, the proposed technique is 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 induces that the proposed method is quite robust, since it doesn't have much dependence on this parameter.

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. And for this critical parameter, no large deviations appear in the results of the proposed method. This further enhances the robustness and reliability of the proposed technique.

Also, 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 (mentioned as IRBF-100) and the classic method for training RBF networks (mentioned as RBF-KMEANS) .And in these experimental results, the reader can see the superiority of the proposed technique over the traditional method of training RBF networks.

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%**
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)**

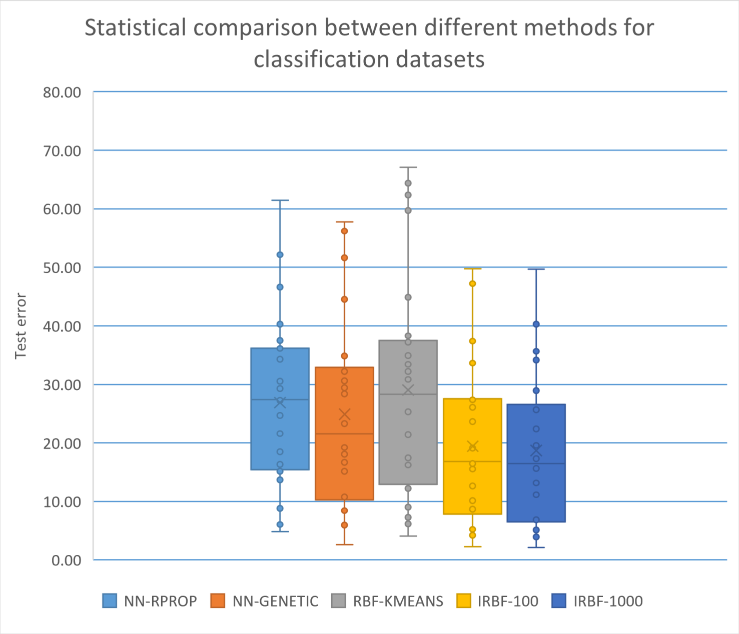

Figure 4. Graphical comparison of all methods for the classification datasets.

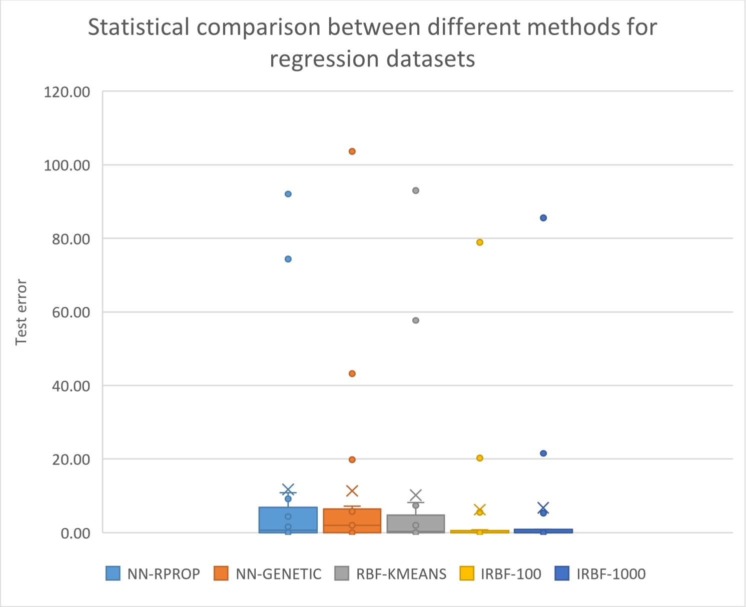

Figure 5. Graphical comparison of the methods for the regression datasets.

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 outperforms 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 its phases. Although, this effect can be overcomed 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 [108]. 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.

Institutional Review Board Statement: Not applicable.

Acknowledgments: The experiments of this research work were performed at the high performance computing system established at 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.

Sample Availability: Not applicable.

References

1. M. Mjahed, The use of clustering techniques for the classification of high energy physics data, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **559**, pp. 199-202, 2006.
2. M Andrews, M Paulini, S Gleyzer, B Poczos, End-to-End Event Classification of High-Energy Physics Data, Journal of Physics: Conference Series **1085**, 2018.
3. P. He, C.J. Xu, Y.Z. Liang, K.T. Fang, Improving the classification accuracy in chemistry via boosting technique, Chemometrics and Intelligent Laboratory Systems 70, pp. 39-46, 2004.
4. J.A. Aguiar, M.L. Gong, T.Tasdizen, Crystallographic prediction from diffraction and chemistry data for higher throughput classification using machine learning, Computational Materials Science **173**, 109409, 2020.
5. I. Kaastra, M. Boyd, Designing a neural network for forecasting financial and economic time series, Neurocomputing **10**, pp. 215-236, 1996.
6. R. Hafezi, J. Shahrabi, E. Hadavandi, A bat-neural network multi-agent system (BNNMAS) for stock price prediction: Case study of DAX stock price, Applied Soft Computing **29**, pp. 196-210, 2015.
7. S.S. Yadav, S.M. Jadhav, Deep convolutional neural network based medical image classification for disease diagnosis. J Big Data **6**, 113, 2019.
8. L. Qing, W. Linhong , D. Xuehai, A Novel Neural Network-Based Method for Medical Text Classification, Future Internet **11**, 255, 2019.
9. J. Park and I. W. Sandberg, Universal Approximation Using Radial-Basis-Function Networks, Neural Computation **3**, pp. 246-257, 1991.
10. J. Ghosh, A. Nag, An Overview of Radial Basis Function Networks. In: Howlett, R.J., Jain, L.C. (eds) Radial Basis Function Networks 2. Studies in Fuzziness and Soft Computing, vol 67. Physica, Heidelberg, 2001.
11. Nam Mai-Duy, Thanh Tran-Cong, Numerical solution of differential equations using multiquadric radial basis function networks, Neural Networks 14, pp. 185-199, 2001.

12. N. Mai-Duy, Solving high order ordinary differential equations with radial basis function networks. Int. J. Numer. Meth. Engng. **62**, pp. 824-852, 2005.
13. C. Laoudias, P. Kemppi and C. G. Panayiotou, Localization Using Radial Basis Function Networks and Signal Strength Fingerprints in WLAN, GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference, Honolulu, HI, 2009, pp. 1-6, 2009.
14. M. Azarbad, S. Hakimi, A. Ebrahimzadeh, Automatic recognition of digital communication signal, International journal of energy, information and communications **3**, pp. 21-33, 2012.
15. P. Teng, Machine-learning quantum mechanics: Solving quantum mechanics problems using radial basis function networks, Phys. Rev. E **98**, 033305, 2018.
16. R. Jovanović, A. Sretenovic, Ensemble of radial basis neural networks with K-means clustering for heating energy consumption prediction, FME Transactions **45**, pp. 51-57, 2017.
17. D.L. Yu, J.B. Gomm, D. Williams, Sensor fault diagnosis in a chemical process via RBF neural networks, Control Engineering Practice **7**, pp. 49-55, 1999.
18. V. Shankar, G.B. Wright, A.L. Fogelson, R.M. Kirby, 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 **75**, pp. 1-22, 2014.
19. W. Shen, X. Guo, C. Wu, D. Wu, Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm, Knowledge-Based Systems 24, pp. 378-385, 2011.
20. J. A. Momoh, S. S. Reddy, Combined Economic and Emission Dispatch using Radial Basis Function, 2014 IEEE PES General Meeting | Conference & Exposition, National Harbor, MD, pp. 1-5, 2014.
21. P. Sohrabi, B. Jodeiri Shokri, H. Dehghani, Predicting coal price using time series methods and combination of radial basis function (RBF) neural network with time series. Miner Econ 2021.
22. U. Ravale, N. Marathe, P. Padiya, Feature Selection Based Hybrid Anomaly Intrusion Detection System Using K Means and RBF Kernel Function, Procedia Computer Science **45**, pp. 428-435, 2015.
23. M. Lopez-Martin, A. Sanchez-Esguevillas, J. I. Arribas, B. Carro, Network Intrusion Detection Based on Extended RBF Neural Network With Offline Reinforcement Learning, IEEE Access **9**, pp. 153153-153170, 2021.
24. H. Yu, T. Xie, S. Paszczynski and B. M. Wilamowski, Advantages of Radial Basis Function Networks for Dynamic System Design,IEEE Transactions on Industrial Electronics v **58**, pp. 5438-5450, 2011.
25. R. Yokota, L.A. Barba, M. G. Knepley, PetRBF — A parallel O(N) algorithm for radial basis function interpolation with Gaussians, Computer Methods in Applied Mechanics and Engineering **199**, pp. 1793-1804, 2010.
26. C. Lu, N. Ma, Z. Wang, Fault detection for hydraulic pump based on chaotic parallel RBF network, EURASIP J. Adv. Signal Process. **2011**, 49, 2011.
27. L.I. Kuncheva, Initializing of an RBF network by a genetic algorithm, Neurocomputing **14**, pp. 273-288, 1997.
28. F. Ros, M. Pintore, A. Deman, J.R. Chrétien, Automatical initialization of RBF neural networks, Chemometrics and Intelligent Laboratory Systems **87**, pp. 26-32, 2007.
29. D. Wang, X.J. Zeng, J.A. Keane, A clustering algorithm for radial basis function neural network initialization, Neurocomputing **77**, pp. 144-155, 2012.
30. E. Ricci, R. Perfetti, Improved pruning strategy for radial basis function networks with dynamic decay adjustment, Neurocomputing **69**, pp. 1728-1732, 2006.
31. Guang-Bin Huang, P. Saratchandran and N. Sundararajan, A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation, IEEE Transactions on Neural Networks **16**, pp. 57-67, 2005.
32. M. Bortman and M. Aladjem, A Growing and Pruning Method for Radial Basis Function Networks, IEEE Transactions on Neural Networks **20**, pp. 1039-1045, 2009.
33. N. B. Karayiannis, M. M. Randolph-Gips, On the construction and training of reformulated radial basis function neural networks, IEEE Transactions on Neural Networks **14**, pp. 835-846, 2003.
34. J.X. Peng, K. Li, D.S. Huang, A Hybrid Forward Algorithm for RBF Neural Network Construction, IEEE Transactions on Neural Networks, **17**, pp. 1439-1451, 2006.
35. D. Du, K. Li, M. Fei, A fast multi-output RBF neural network construction method, Neurocomputing **73**, pp. 2196-2202, 2010.
36. V. Agarwal, S. Bhanot, Radial basis function neural network-based face recognition using firefly algorithm, Neural Comput & Applic **30**, pp. 2643–2660, 2018.
37. S. Jiang et al., Prediction of Ecological Pressure on Resource-Based Cities Based on an RBF Neural Network Optimized by an Improved ABC Algorithm, IEEE Access. **7**, pp. 47423-47436, 2019.
38. F. Marini, B. Walczak, Particle swarm optimization (PSO). A tutorial, Chemometrics and Intelligent Laboratory Systems **149**, pp. 153-165, 2015.
39. B. Liu, L. Wang, Y.H. Jin, An Effective PSO-Based Memetic Algorithm for Flow Shop Scheduling, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **37**, pp. 18-27, 2007.
40. J. Yang, L. He, S. Fu, An improved PSO-based charging strategy of electric vehicles in electrical distribution grid, Applied Energy **128**, pp. 82-92, 2014.
41. K. Mistry, L. Zhang, S. C. Neoh, C. P. Lim, B. Fielding, A Micro-GA Embedded PSO Feature Selection Approach to Intelligent Facial Emotion Recognition, IEEE Transactions on Cybernetics. **47**, pp. 1496-1509, 2017.

42. S. Han, X. Shan, J. Fu, W. Xu, H. Mi, Industrial robot trajectory planning based on improved pso algorithm, J. Phys.: Conf. Ser. **1820**, 012185, 2021.
43. C.A. Floudas, C.E. Gounaris, A review of recent advances in global optimization, J Glob Optim **45**, pp. 3–38, 2009.
44. H. Wang, W. Wang, X. Zhou, H. Sun, J. Zhao, X. Yu, Z. Cui, Firefly algorithm with neighborhood attraction, Information Sciences **382-383**, pp. 374-387, 2017.
45. D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing Company, Reading, Massachussets, 1989.
46. Z. Michaelewicz, Genetic Algorithms + Data Structures = Evolution Programs. Springer - Verlag, Berlin, 1996.
47. S.A. Grady, M.Y. Hussaini, M.M. Abdullah, Placement of wind turbines using genetic algorithms, Renewable Energy **30**, pp. 259-270, 2005.
48. I.U. Khan,N. Aslam, R. Alshehri, S. Alzahrani, M. Alghamdi, A. Almalki, M. Balabeed, Cervical Cancer Diagnosis Model Using Extreme Gradient Boosting and Bioinspired Firefly Optimization, Scientific Programming **2021**, Article ID 5540024, 2021.
49. M. Zivkovic, N. Bacanin, M. Antonijevic, B. Nikolic, G. Kvascev, M. Marjanovic, N. Savanovic, Hybrid CNN and XGBoost Model Tuned by Modified Arithmetic Optimization Algorithm for COVID-19 Early Diagnostics from X-ray Images, Electronics **11**, 3798, 2022.
50. J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, Vol. 1, No. 14, pp. 281-297, 1967.
51. E. Hansen and G.W. Walster.Global Optimization using Interval Analy-sis.Marcel Dekker Inc., New York, 2004.
52. M. Markót, J. Fernández, L. Casado et al, New interval methods for constrained global optimization, Mathematical Programming **106**, pp. 287-318, 2006.
53. A. Žilinskas, J. Žilinskas, Interval Arithmetic Based Optimization in Nonlinear Regression, Informatica **21**, pp. 149-158, 2010.
54. C.A. Schnepper, M.A. Stadtherr, Robust process simulation using interval methods, Computers & Chemical Engineering **20**, pp. 187-199, 1996.
55. C. Carreras, I. D. Walker, Interval methods for fault-tree analysis in robotics, IEEE Transactions on Reliability **50**, pp. 3-11, 2001.
56. A. Serguieva, J. Hunte, Fuzzy interval methods in investment risk appraisal, Fuzzy Sets and Systems **142**, pp. 443-466, 2004.
57. Riccardo Poli, James Kennedy kennedy, Tim Blackwell, Particle swarm optimization An Overview, Swarm Intelligence **1**, pp 33-57, 2007.
58. I.C. Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Information Processing Letters **85**, pp. 317-325, 2003.
59. Y. Shi, R.C. Eberhart, Empirical study of particle swarm optimization, In: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), pp. 1945-1950 Vol. 3, 1999.
60. B. Borowska, Exponential Inertia Weight in Particle Swarm Optimization. In: Wilimowska, Z., Borzemski, L., Grzech, A., Świątek, J. (eds) Information Systems Architecture and Technology: Proceedings of 37th International Conference on Information Systems Architecture and Technology – ISAT 2016 – Part IV. Advances in Intelligent Systems and Computing, vol 524. Springer, Cham, 2017.
61. L. Zhang, H. Yu, S. Hu, A New Approach to Improve Particle Swarm Optimization. In: , et al. Genetic and Evolutionary Computation — GECCO 2003. GECCO 2003. Lecture Notes in Computer Science, vol 2723. Springer, Berlin, Heidelberg, 2003.
62. B. Borowska, Dynamic Inertia Weight in Particle Swarm Optimization. In: Shakhovska, N., Stepashko, V. (eds) Advances in Intelligent Systems and Computing II. CSIT 2017. Advances in Intelligent Systems and Computing, vol 689 . Springer, Cham, 2018.
63. Y. Shi, R.C. Eberhart, Fuzzy adaptive particle swarm optimization, In: Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546), pp. 101-106 vol. 1, 2001.
64. P. Kaelo, M.M. Ali, Integrated crossover rules in real coded genetic algorithms, European Journal of Operational Research **176**, pp. 60-76, 2007.
65. I.G. Tsoulos, Modifications of real code genetic algorithm for global optimization, Applied Mathematics and Computation **203**, pp. 598-607, 2008.
66. J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera. KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. Journal of Multiple-Valued Logic and Soft Computing 17, pp. 255-287, 2011.
67. Weiss, Sholom M. and Kulikowski, Casimir A., Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems, Morgan Kaufmann Publishers Inc, 1991.
68. J.R. Quinlan, Simplifying Decision Trees. International Journal of Man-Machine Studies **27**, pp. 221-234, 1987.
69. T. Shultz, D. Mareschal, W. Schmidt, Modeling Cognitive Development on Balance Scale Phenomena, Machine Learning **16**, pp. 59-88, 1994.
70. Z.H. Zhou,Y. Jiang, NeC4.5: neural ensemble based C4.5," in IEEE Transactions on Knowledge and Data Engineering **16**, pp. 770-773, 2004.
71. R. Setiono , W.K. Leow, FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks, Applied Intelligence **12**, pp. 15-25, 2000.
72. B. Evans, D. Fisher, Overcoming process delays with decision tree induction, IEEE Expert **9**, pp. 60-66, 1994.

73. G. Demiroz, H.A. Govenir, N. Ilter, Learning Differential Diagnosis of Eryhemato-Squamous Diseases using Voting Feature Intervals, Artificial Intelligence in Medicine. **13**, pp. 147–165, 1998.
74. B. Hayes-Roth, B., F. Hayes-Roth. Concept learning and the recognition and classification of exemplars. Journal of Verbal Learning and Verbal Behavior **16**, pp. 321-338, 1977.
75. I. Kononenko, E. Šimec, M. Robnik-Šikonja, Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF, Applied Intelligence **7**, pp. 39–55, 1997
76. R.M. French, N. Chater, Using noise to compute error surfaces in connectionist networks: a novel means of reducing catastrophic forgetting, Neural Comput. **14**, pp. 1755-1769, 2002.
77. J.G. Dy , C.E. Brodley, Feature Selection for Unsupervised Learning, The Journal of Machine Learning Research **5**, pp 845–889, 2004.
78. S. J. Perantonis, V. Virvilis, Input Feature Extraction for Multilayered Perceptrons Using Supervised Principal Component Analysis, Neural Processing Letters **10**, pp 243–252, 1999.
79. J. Garcke, M. Griebel, Classification with sparse grids using simplicial basis functions, Intell. Data Anal. **6**, pp. 483-502, 2002.
80. G. Cestnik, I. Konenenko, I. Bratko, Assistant-86: A Knowledge-Elicitation Tool for Sophisticated Users. In: Bratko, I. and Lavrac, N., Eds., Progress in Machine Learning, Sigma Press, Wilmslow, pp. 31-45, 1987.
81. M. Elter, R. Schulz-Wendtland, T. Wittenberg, The prediction of breast cancer biopsy outcomes using two CAD approaches that both emphasize an intelligible decision process, Med Phys. **34**, pp. 4164-72, 2007.
82. Little MA, McSharry PE, Hunter EJ, Spielman J, Ramig LO. Suitability of dysphonia measurements for telemonitoring of Parkinson's disease. IEEE Trans Biomed Eng. 2009;56(4):1015. doi:10.1109/TBME.2008.2005954
83. J.W. Smith, J.E. Everhart, W.C. Dickson, W.C. Knowler, R.S. Johannes, 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, pp.261-265, 1988.
84. D.D. Lucas, R. Klein, J. Tannahill, D. Ivanova, S. Brandon, D. Domyancic, Y. Zhang, Failure analysis of parameter-induced simulation crashes in climate models, Geoscientific Model Development **6**, pp. 1157-1171, 2013.
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 (2015) Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS, 2015-November, art. no. 7319047, pp. 3097-3100.
86. T. Hastie, R. Tibshirani, Non-parametric logistic and proportional odds regression, JRSS-C (Applied Statistics) **36**, pp. 260–276, 1987.
87. M. Dash, H. Liu, P. Scheuermann, K. L. Tan, Fast hierarchical clustering and its validation, Data & Knowledge Engineering **44**, pp 109–138, 2003.
88. W.H. Wolberg, O.L. Mangasarian, Multisurface method of pattern separation for medical diagnosis applied to breast cytology, Proc Natl Acad Sci U S A. **87**, pp. 9193–9196, 1990.
89. M. Raymer, T.E. Doom, L.A. Kuhn, W.F. Punch, Knowledge discovery in medical and biological datasets using a hybrid Bayes classifier/evolutionary algorithm. IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society, **33** , pp. 802-813, 2003.
90. P. Zhong, M. Fukushima, Regularized nonsmooth Newton method for multi-class support vector machines, Optimization Methods and Software **22**, pp. 225-236, 2007.
91. R.G. Andrzejak, K. Lehnertz, F. Mormann, C. Rieke, P. David, and C. E. Elger, 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 **64**, pp. 1-8, 2001.
92. M. Koivisto, K. Sood, Exact Bayesian Structure Discovery in Bayesian Networks, The Journal of Machine Learning Research **5**, pp. 549–573, 2004.
93. W. J Nash, T.L. Sellers, S.R. Talbot, A.J. Cawthor, W.B. Ford, 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 (ISSN 1034-3288), 1994.
94. T.F. Brooks, D.S. Pope, and A.M. Marcolini. Airfoil self-noise and prediction. Technical report, NASA RP-1218, July 1989.
95. J.S. Simonoff, Smooting Methods in Statistics, Springer - Verlag, 1996.
96. I.Cheng Yeh, Modeling of strength of high performance concrete using artificial neural networks, Cement and Concrete Research. **28**, pp. 1797-1808, 1998.
97. D. Harrison and D.L. Rubinfeld, Hedonic prices and the demand for clean ai, J. Environ. Economics & Management **5**, pp. 81-102, 1978.
98. J.S. Simonoff, Smooting Methods in Statistics, Springer - Verlag, 1996.
99. Mackowiak, P.A., Wasserman, S.S., Levine, M.M., 1992. 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. 268, 1578–1580
100. R.D. King, S. Muggleton, R. Lewis, M.J.E. Sternberg, Proc. Nat. Acad. Sci. USA **89**, pp. 11322–11326, 1992.
101. M. Sikora, L. Wrobel, Application of rule induction algorithms for analysis of data collected by seismic hazard monitoring systems in coal mines, Archives of Mining Sciences **55**, pp. 91-114, 2010.

102. C. Sanderson, R. Curtin, Armadillo: a template-based C++ library for linear algebra, Journal of Open Source Software **1**, pp. 26, 2016.
103. R. Chandra, L. Dagum, D. Kohr, D. Maydan,J. McDonald and R. Menon, Parallel Programming in OpenMP, Morgan Kaufmann Publishers Inc., 2001.
104. M. Riedmiller and H. Braun, A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP algorithm, Proc. of the IEEE Intl. Conf. on Neural Networks, San Francisco, CA, pp. 586–591, 1993.
105. C. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.
106. G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of Control Signals and Systems **2**, pp. 303-314, 1989.
107. Grzegorz Klima, Fast Compressed Neural Networks, available from http://fcnn.sourceforge.net/.
108. S. Das, P. N. Suganthan, Differential Evolution: A Survey of the State-of-the-Art, IEEE Transactions on Evolutionary Computation **15**, pp. 4-31, 2011.