MDPI

Article

From Initialization to Convergence: A Three-Stage technique for Robust RBF Network Training

Ioannis G. Tsoulos[1,*], Vasileios Charilogis[2], Dimitrios Tsalikakis[3]

[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
[3] Department of Engineering Informatics and Telecommunications, University of Western Macedonia, 50100 Kozani, Greece; dtsalikakis@uowm.gr
* Correspondence: itsoulos@uoi.gr

Abstract

A parametric machine learning tool with many applications is the Radial Basis Function (RBF) network, that have been incorporated in various classification or regression problems. A key component of these networks is their radial functions. These networks acquire adaptive capabilities through a two-stage training technique in most cases. During the first stage, the parameters of the model (called centers and variances) are estimated, and in the second stage, that involves solving a linear system of equations, the external weights for the radial functions are adjusted. However, in many cases, this training technique has reduced performance either due to instability in arithmetic operations or due to the trapping in local minima of the training error. In this paper, a three-stage method is suggested in this work, to address the above problems. In the first stage, an initial estimate of the intervals for the network parameter values is made, in the second stage, the network parameter values are adjusted within the intervals of the first phase, and finally, in the third stage of the proposed technique, a local optimization method is used for the final adjustment of the network parameters. The proposed method was tested against some machine learning models from the related literature as well as the original RBF training method on a wide series of classification and regression datasets, found in online databases, and a comparison was made in terms of classification or regression error. It should be noted that although the proposed methodology had very good results in the above measurements, it requires significant computational execution time, due to the use of three phases of processing and adaptation of the network parameters.

Keywords: Machine learning; Neural networks; Genetic algorithms; Optimization

Received:
Revised:
Accepted:
Published:

Citation: Tsoulos, I.G.; Charilogis, V.; Tsalikakis D. From Initialization to Convergence: A Three-Stage technique for Robust RBF Network Training. *Journal Not Specified* **2025**, *1*, 0. https://doi.org/

1. Introduction

Many practical problems can be handled by machine learning tools. These problems derived from areas, such as physics [1,2], astronomy [3,4], chemistry [5,6], medicine [7,8], economics [9,10], image processing [11], time series forecasting [12] etc. Among the most used machine learning tools, one can detect the Radial Basis Function (RBF) network. These networks are commonly defined by the following equation:

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

The symbols appeared in this equation are defined as follows:

1. The vector $\overrightarrow{x}$ stands for the input pattern. The dimension of each pattern is denoted by d.
2. The number of weights for the network are expressed through the variable k. The vector $\overrightarrow{w}$ denotes these weights.
3. The vectors $\overrightarrow{c_i}$, $i = 1, .., k$ denote the centers for the functions used in the network.
4. The function $\phi(x)$ denotes a Gaussian function defined as:

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

An example plot for the Gaussian function with the following set of parameters: $c = 0$, $\sigma = 1$ is depicted in Figure 1.

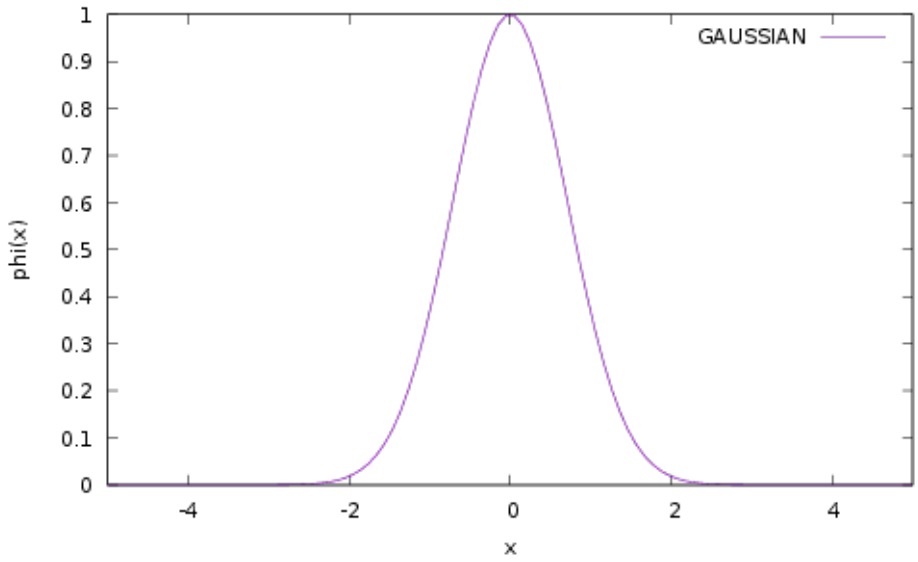

Figure 1. An example plot for the Gaussian function with $\sigma = 1$ and $c = 0$.

From the graph above, one can conclude that the value of the function rapidly approaches zero as the value of the variable x moves away from the center c. The training error of any given $R(x)$ RBF network is defined as:

$$E\left(R(\overrightarrow{x})\right) = \sum_{i=1}^{M}\left(R(\overrightarrow{x_i}) - y_i\right)^2 \tag{3}$$

The set $(\overrightarrow{x_i}, y_i)$, $i = 1, ..., M$ denotes the training set of the objective problem and the values y_i are considered as the actual output for each pattern $\overrightarrow{x_i}$.

A magnitude of problems from the real world can be tackled using RBF networks, such as face recognition [13], numerical problems [14,15], economic problems [16], robotics [17,18], network security [19,20], classification of process faults [21], time series prediction [22], estimation of wind power production [23] etc. Moreover, Park et al. [25] proved that an RBF network with one processing layer can be approximate any given function.

In this paper, the use of a three-stage technique is suggested for the effective training of RBF networks. In the first stage of the technique, the range of values for the parameters of the RBF network is detected. This detection is implemented using the K-Means algorithm [24] for the weights and the variances of the radial functions. After applying the above procedure, a range of values for the network parameters is created which directly depends on the values produced by K-means algorithm. During the second stage of the proposed work, a global optimization procedure is incorporated to optimize the parameters of the RBF

network with respect to the equation 3. The training of the parameters is performed inside the interval of values created during the first stage of the technique. In the current work, the Genetic Algorithm is used as the method of the second phase, but any optimization technique can be incorporated. Finally, in the third stage of the proposed work, a local optimization procedure is applied to the best solution located in the second phase. The purpose of the present technique is first to identify a reliable range of values for the parameters of RBF networks and then to train the network parameters within this range of values, avoiding possible arithmetic instability problems presented by the established method of training RBF networks.

The proposed technique is the application of three distinct procedures in series, where in each stage the results of the previous phase are used. In the first phase, an initial estimation of the centers and variances is performed with the method of K-Means. This method is preferred because it is extremely fast to execute and can provide an overview of the search space for neural networks. It is preferred over using random values as this would require a significantly large number of iterations for proper initialization of the parameters. After executing the K-Means technique, a range of values for the network parameters is created with values that are multiples of those produced by the K-Means technique. In this way, on the one hand, the use of the K-Means technique is utilized and, on the other hand, the second-phase optimization algorithm is given the opportunity to search for parameter values that yield lower values of the error function close to the initial values of the first phase. A genetic algorithm is used as the optimization algorithm in the second phase due to its adaptability to any environment and their widespread use in many computational problems. However, any universal optimization technique could be utilized in this phase. However, although genetic algorithms are extremely effective methods of global optimization, they often do not locate exactly a true minimum of the objective function and therefore the help of a local minimization method is deemed necessary. This occurs in the third phase, where a local optimization method is applied to the chromosome with the smallest value produced in the second phase. The minimum found in this phase is also the final result of the algorithm, which is also a configuration of the parameters of the RBF network.

The main characteristics of the present technique are that it uses different techniques in series in order to identify the optimal set of parameters for the machine learning model and to avoid possible overfitting problems. In the first discrete phase, a well-established clustering technique such as the K-Means technique is used to identify an initial range of values for the model parameters. In the second stage, a global optimization algorithm is used to minimize the error function considering the range of values of the first phase and in the last phase a local minimization method will be used to find a guaranteed local minimum of the error function. The method of the last phase guarantees the training of the parameters within the range of values of the first phase.

The remaining of this manuscript has the following structure: Section 2 discusses the used method and presents the associated algorithms, section 3 presents the datasets used in the conducted experiments as well as the experimental results and finally 4 presents some conclusions.

2. Materials and Methods

The three distinct phases of the proposed method are discussed here. The discussion initiates with the first phase, where the construction of the ranges for the parameter values is performed using the K-means algorithm. Subsequently, the steps of the used Genetic Algorithm are presented in detail and finally, this section concludes with the description of the final phase, where a local optimization method is applied to the best

located chromosome of the second phase. Also, the description of the used datasets is provided in this section in table format.

2.1. Related work

A number of techniques have been published during the past years for the efficient training of neural networks and the efficient adaptation of parameters. Among them, one can find methods aimed to efficient initialize the parameters of RBF networks [26–28]. Moreover, Benoudjit et al. provided a discussion on the estimation of kernel widths in these models [29]. Additionally, a series of pruning techniques [30–32] have been introduced aiming to reduce the required number of parameters for networks, providing a solution to the overfitting problem. Also, methods that construct the architecture of RBF networks have been proposed in the recent literature, such as the work of Du et al. [33] or the work of Yu et al. who suggested an incremental design framework for RBF networks [34]. Also, a series of optimization techniques have been used in the past for the minimization of equation 3, such as genetic algorithms [35,36], the Particle Swarm Optimization method [37,38], the Differential Evolution technique [39] etc. Furthermore, the rapid increase in the use of parallel computing techniques in recent decades has resulted in the publication of a series of relevant scientific papers that exploit these techniques [40,41]. Recently, Rani et al. proposed an improved PSO optimizer that integrates a differential search mechanism for efficient RBF network training [42]. Moreover, Karamichailidou et al. suggested a novel method for RBF training using variable projection and fuzzy means [43].

2.2. The description of the first phase

The method of K-means, used widely in machine learning, is incorporated in the first phase for the location of the ranges for the centers and variances of the model. This method is incorporated to locate the centers and the variances of the possible groups of a series of points. Furthermore, a series of extensions of this method have been published during the past years, such as the Genetic K-means algorithm [44], the unsupervised K-means algorithm [45], the Fixed-centered K-means algorithm [46] etc. A detailed review of the K-means method can be located in the work of Oti et al. [47]. The K-means method is presented in Algorithm 1. A graphical rerpesentation is provided in Figure 2.

Algorithm 1 A detailed description of K-means algorithm.

1. **Input:** The set of patterns of the objective problem $(\overrightarrow{x_i}),\ i = 1, ..., M$
2. **Input**: the number of centers k.
3. **Output**: The vectors $\overrightarrow{c_i},\ i = 1, .., k$ and the quantities $\sigma_i,\ i = 1, \ldots, k$
4. **Set** $S_j = \{\},\ j = 1..k$, as the sets of samples belonging to the same group.
5. **For** every pattern $x_i,\ i = 1, ..., M$ **do**
   (a) **Set** $j^* = \min_{i=1}^{k}\{D(x_i, c_j)\}$.
   (b) **Set** $S_{j^*} = S_{j^*} \cup \{x_i\}$.
6. **EndFor**
7. **For** each center $c_j,\ j = 1..k$ **do**
   (a) **Denote** with M_j the total number of patterns that have been assigned to cluster S_j
   (b) **Compute** c_j as

$$c_j = \frac{1}{M_j}\sum_{i=1}^{M_j} x_i$$

8. **EndFor**
9. **Calculate** the quantities s_j as

$$\sigma_j^2 = \frac{\sum_{i=1}^{M_j}(x_i - c_j)^2}{M_j}$$

10. The algorithm **terminates** when there are no changes in the estimated centers c_j.
11. **Go to** step 5.

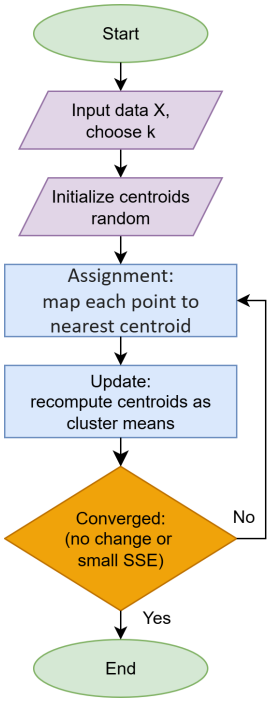

Figure 2. A graphical presentation of the K-means algorithm.

After the calculation of $\overrightarrow{c_i},\ i = 1, .., k$ and the quantities $\sigma_i,\ i = 1, \ldots, k$, the method locates the bound vectors $\overrightarrow{L}$, $\overrightarrow{R}$ for the parameters of the RBF network. The dimension of the bound vectors is defined as:

$$n = (d + 2) \times k \tag{4}$$

The algorithm shown graphically in Figure 3 describes a procedure for computing safe parameter bounds informed by K-means. The process begins by preparing the inputs the cluster centroids and spreads, a positive initial bound for the weights, and a scaling factor $F \geq 1$ aiming to produce the lower and upper bound vectors L and R. For each group and for each of its features, symmetric bounds around zero are computed as F times the corresponding centroid coordinate (i.e., $L_m = -F \times c_{ij},\ R_m = F \times c_{ij}$). After completing a group's features, a complementary step computes bounds based on that group's spread, using $F \times \sigma_i$ (i.e., $L_m = -F \times \sigma_i,\ R_m = F \times \sigma_i$). In this way, every group receives a complete, consistent specification through two contributions: one driven by feature centroids and one capturing group extent. Once all groups have been processed, initial bounds for the combining weights are assigned using a fixed symmetric limit B_w (i.e., $L_m = -B_w,\ R_m = B_w$). Finally, all computed limits are assembled into L and R, ready for training, estimation, or validity checks. The overall logic is hierarchical and layered: leveraging centroid information, incorporating spread, and bounding the weights, with clear and reproducible steps throughout the flow.

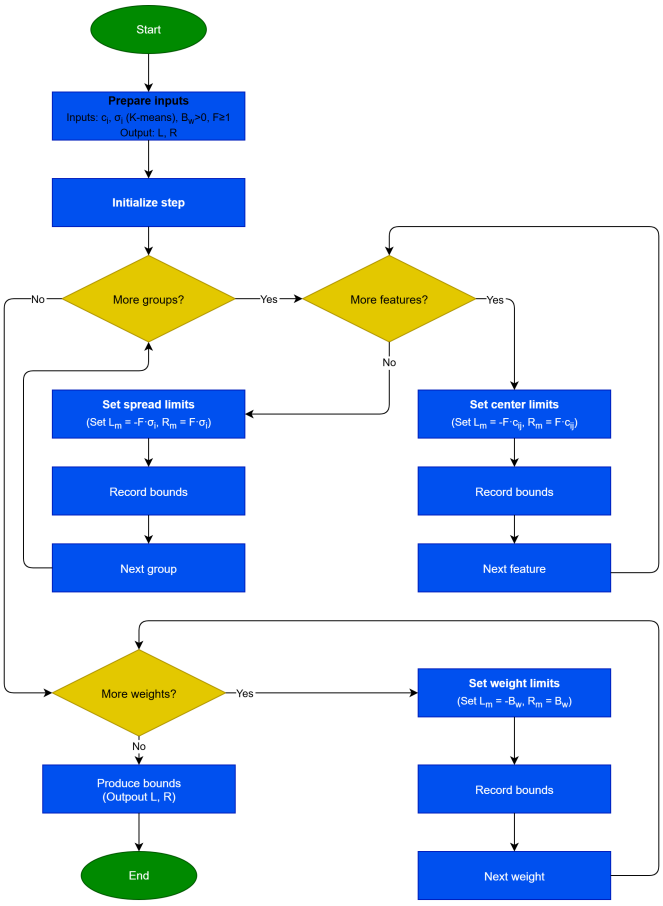

Figure 3. The bound construction algorithm.

2.3. The description for the second phase

During the second phase, an optimization procedure is utilized to minimize the equation 3 inside the bound vectors $\overrightarrow{L}$, $\overrightarrow{R}$ of the previous phase.In the proposed implementation, the Genetic Algorithm was incorporated during the second phase. Genetic algorithms are evolutionary methods that are based on randomly produced solutions of the objective problem. The trial solutions of the genetic algorithm are commonly called chromosomes, and they are evolved through some operations that are similar to natural processes, such as selection, crossover and mutation. They have been incorporated in a wide series of problems, such as energy problems [48], water distribution [49], problems

appearing in banking transactions [50], optimization of neural networks [51] etc. Also, another advantage of Genetic Algorithms is that they can easily adopt parallel programming techniques in order to speed up the evolutionary process [52,53]. The layout of the chromosomes used in the obtained genetic algorithm is presented in Figure 4. In this layout the following assumptions hold:

1. The value $c_{i,j}$ denotes the j element of the i center of the RBF network, with $i \in [1,k]$ and $j \in [1,d]$.
2. The value σ_i represents the σ parameter for the corresponding radial function.
3. The value w_i, $i \in [1,k]$ represents the weight for the corresponding radial function.

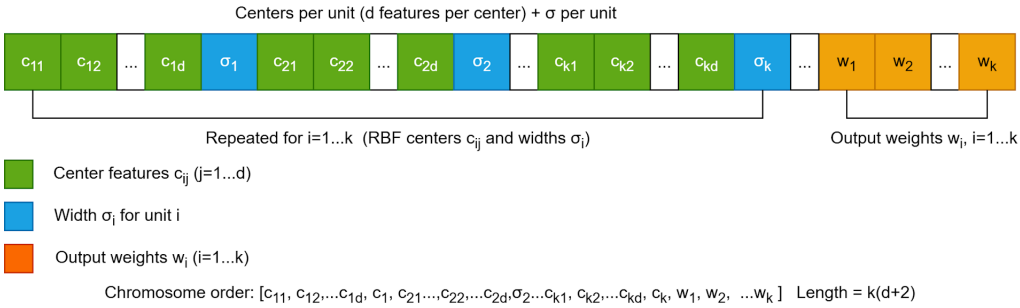

Figure 4. The layout of chromosomes used in the second stage of the proposed method.

The steps of algorithm used in the second phase are the following:

1. **Initialization step**.
   (a) The following set of parameter are **initialized:** N_c for the number of chromosomes, N_g used to express the maximum number of allowed generations, p_s that defines the selection rate and finally p_m which is used to represent the mutation rate.
   (b) **Perform** the initialization of each chromosome g_i, $i = 1, \ldots, N_c$ in the population. Each chromosome is considered as a set of randomly selected integers following the scheme of Figure 4 and the initialization is performed inside the bound vectors $\overrightarrow{L}$, $\overrightarrow{R}$.
   (c) **Set** $k = 0$. This variable denotes the number of generations.
2. **Fitness calculation step**.
   (a) **For** $i = 1, \ldots, N_c$ **do**
      i. **Produce** an RBF network $R_i = R(\overrightarrow{x}, \overrightarrow{g_i})$. The parameters of this network are stored in the chromosome $\overrightarrow{g_i}$.
      ii. **Estimate** the related fitness f_i as

$$f_i = \sum_{j=1}^{M} \left(R(\overrightarrow{x}_j, \overrightarrow{g_i}) - y_j \right)^2 \tag{5}$$

   (b) **End For**
3. **Genetic operations step**.
   (a) Selection procedure: Initially a sorting of all chromosomes is performed according to their fitness values. The first $p_s \times N_c$ of them are transferred without any change to the next generation. The remaining chromosomes will be substituted by offrpsings produced using the operations of crossover and mutation.
   (b) Crossover procedure: During this procedure In this procedure $(1 - p_s) \times N_c$ new chromosomes will be constructed. For each pair $(\tilde{z}, \tilde{w})$ of new chromo-

somes, two chromosomes (z, w) are chosen using the procedure of tournament selection. The new offsprings are produced following the scheme:

$$\begin{aligned} \tilde{z}_i &= a_i z_i + (1 - a_i) w_i \\ \tilde{w}_i &= a_i w_i + (1 - a_i) z_i \end{aligned} \tag{6}$$

where a_i are random numbers and the following holds: $a_i \in [-0.5, 1.5]$ [54].

(c) Mutation procedure: A random number $r \in [0, 1]$ is picked for each element $t_j, j = 1, \ldots, n$ and for every chromosome g_i . If $r \leq p_m$, then this element is altered according to the following equation:

$$t'_j = \begin{cases} t_j + \Delta(k, R_j - t_j), & t = 0 \\ t_j - \Delta(k, t_j - L_j), & t = 1 \end{cases} \tag{7}$$

The value t is a random number that can be either 0 or 1. The function function $\Delta(k, y)$ is provided by the following equation:

$$\Delta(k, y) = y\left(1 - r^{\left(1 - \frac{k}{N_g}\right)}\right) \tag{8}$$

4. **Termination check step**.
   (a) **Set** $k = k + 1$
   (b) **If** $k < N_g$ then go to Fitness Calculation Step
   (c) **Else return** the best located chromosome g^* as the outcome of this algorithm.

2.4. The steps of the third phase

In the third phase of the proposed method, a local optimization procedure is applied to the outcome of the previous phase, in order to locate an actual minimum for the training error of the RBF network. In this work, a BFGS variant of Powell [55] was utilized as the local search procedure. This variant can preserve the bounds located previously in an efficient way. During the past years a series of modifications to the BFGS method have been introduced, such as the limited memory variant L-BFGS ideal for large scale problems [56] or the Regularized Stochastic BFGS Algorithm [57]. Also, Dai published an article on the convergence properties of the BFGS method [58]. The main steps of the final phase of the algorithm are:

1. **Get** the best chromosome $\overrightarrow{g^*}$ of the previous phase.
2. **Produce** the RBF network that corresponds to this chromosome and denoted as $R^* = R\left(\overrightarrow{x}, \overrightarrow{g^*}\right)$.
3. **Minimize** the training error of the network R^* using the local search procedure of this phase.
4. The final model is **applied** to the test set and the corresponding test error is calculated and reported.

A summary flow chart showing the sequence of the various phases of the proposed work is presented in Figure 5.

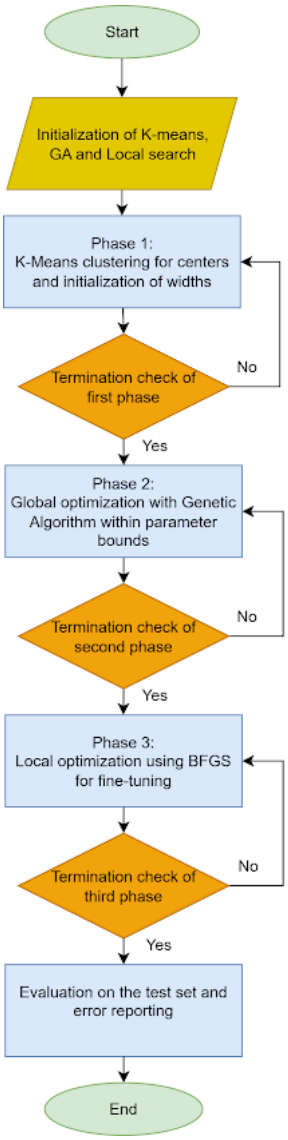

Figure 5. Summary flowchart of the proposed method.

2.5. The experimental datasets.

The proposed method was tested on a wide series of classification and regression problems, that can be downloaded UCI database [59], the KEEL database [60] and the STATLIB database [61]. The classification datasets used in the experiments accompanied with their information (patterns and number of classes) are presented in Table 1.

Table 1. The column DATASET represents the name of the dataset. The column entitled Reference Paper denotes the paper, where this dataset was mentioned. The column Patterns stands for the number of patterns in the dataset and the column Number of Classes represents the number of distinct classes in the mentioned dataset.

DATASET	Reference paper	Patterns	Number of Classes
APPENDICITIS	[62]	106	2
ALCOHOL	[63]	476	4
AUSTRALIAN	[64]	690	2
BALANCE	[65]	625	3
CLEVELAND	[66,67]	297	5
CIRCULAR	[68]	500	2
DERMATOLOGY	[69]	368	6
ECOLI	[70]	336	8
HAYES ROTH	[71]	132	3
HEART	[72]	270	2
HEARTATTACK	[73]	303	2
HOUSEVOTES	[74]	232	2
IONOSPHERE	[75,76]	351	2
LIVERDISORDER	[77,78]	345	2
LYMOGRAPHY	[79]	148	4
MAMMOGRAPHIC	[80]	830	2
PARKINSONS	[81,82]	195	2
PIMA	[83]	768	2
POPFAILURES	[84]	540	2
REGIONS2	[85]	622	5
SAHEART	[86]	462	2
SEGMENT	[87]	2300	7
SPIRAL	[68]	2000	2
STATHEART	[88]	270	2
STUDENT	[89]	403	2
TRANSFUSION	[90]	748	2
WDBC	[91,92]	569	2
WINE	[93,94]	179	3
Z_F_S	[95,96]	300	3
Z_O_N_F_S	[95,96]	500	5
ZO_NF_S	[95,96]	500	3
ZONF_S	[95,96]	500	2
ZOO	[97]	101	7

Also, table 2 presents the used regression datasets that was incorporated in the conducted experiments.

Table 2. The list of regression datasets.

DATASET	Reference paper	Patterns
ABALONE	[98]	4177
AIRFOIL	[99]	1483
AUTO	[100]	392
BK	[68]	96
BL	[68]	43
BASEBALL	[60]	337
CONCRETE	[101]	1030
DEE	[60]	365
FA	[61]	252
FRIEDMAN	[102]	1200
FY	[61]	125
HO	[61]	506
HOUSING	[103]	506
LASER	[59]	993
LW	[61]	189
MORTGAGE	[60]	1049
PL	[61]	1650
PLASTIC	[59]	1670
QUAKE	[59]	2178
SN	[61]	576
STOCK	[60]	950
TREASURY	[60]	1049

3. Results

3.1. Experimental results

The computation environment for the experiments was a Debian Linux system equipped with 128GB of RAM and all the necessary code was written in the C++ programming language. Also, the OPTIMUS OPTIMUS optimization environment [104] available from https://github.com/itsoulos/GlobalOptimus.git (accessed on 9 October 2025) was used for the optimization methods. The validation of the experiments was performed using the ten - fold cross validation technique. For the classification datasets, the average classification error, calculated on the corresponding test set, is reported. This quantity is expressed using the following equation:

$$E_C\left(N\left(\overrightarrow{x},\overrightarrow{w}\right)\right) = 100 \times \frac{\sum_{i=1}^{N}\left(\text{class}\left(N\left(\overrightarrow{x_i},\overrightarrow{w}\right)\right) - y_i\right)}{N} \tag{9}$$

The set T represents the associated test set, where $T = (x_i, y_i)$, $i = 1, \ldots, N$. Similarly, for the regression datasets, the average regression error is defined as follows:

$$E_R\left(N\left(\overrightarrow{x},\overrightarrow{w}\right)\right) = \frac{\sum_{i=1}^{N}\left(N\left(\overrightarrow{x_i},\overrightarrow{w}\right) - y_i\right)^2}{N} \tag{10}$$

Table 3 contains the values for each parameter of the current method.

Table 3. The values for each parameter of the proposed method.

NAME	MEANING	VALUE
k	Number of radial functions	10
F	Scaling factor	2.0
B_w	Bound value for the weights	10.0
N_c	Number of chromosomes	500
N_g	Maximum number of generations	200
p_s	Selection rate	0.1
p_m	Mutation rate	0.05

In tables that contain the experimental results, the following notation is used:

1. The column DATASET is used to represent the name of the used dataset.
2. The results from the incorporation of the BFGS procedure [105] to train an artificial neural network [106,107] with 10 weights are presented in column under the title BFGS.
3. The results from the usage of the ADAM optimization method [108,109] to train an artificial neural network with 10 weights are presented in the column ADAM.
4. The column RBF-KMEANS denotes the incorporation of the original training method of RBF networks to train an RBF network with 10 nodes.
5. The column NEAT (NeuroEvolution of Augmenting Topologies) [110] stands for the method NEAT incorporated in the training of neural networks.
6. The column DNN stands for the incorporation of a deep neural network as implemented in the Tiny Dnn library, that can be downloaded freely from https://github.com/tiny-dnn/tiny-dnn(accessed on 9 October 2025). The network was trained with the AdaGrad optimizer [111].
7. The column BAYES stands for the usage of the Bayesian optimizer as implemented in the freely available library BayesOpt [112], used to train a neural network with 10 processing nodes. The code is available from https://github.com/rmcantin/bayesopt (accessed on 9 October 2025).
8. The column GENRBF stands method introduced in [113] for RBF training.
9. The column PROPOSED is used to represent the results obtained by the current work.
10. The row average is used to hold the average regression or classification error for all datasets.
11. The bold notation is used in all experimental tables to indicate the machine learning method with the lowest classification or rergression error.

Table 4. Experimental results on the classification datasets with the machine learning methods mentioned in this section.

DATASET	BFGS	ADAM	NEAT	DNN	BAYES	RBF-KMEANS	GENRBF	PROPOSED
Alcohol	41.50%	57.78%	66.80%	39.04%	30.85%	49.38%	52.45%	**28.57%**
Appendicitis	18.00%	16.50%	17.20%	17.30%	15.00%	**12.23%**	16.83%	15.00%
Australian	38.13%	35.65%	31.98%	35.03%	34.83%	34.89%	41.79%	**22.67%**
Balance	8.64%	**7.87%**	23.14%	24.56%	8.13%	33.42%	38.02%	13.11%
Cleveland	77.55%	67.55%	53.44%	63.28%	64.79%	67.10%	67.47%	**50.86%**
Circular	6.08%	19.95%	35.18%	21.87%	21.06%	5.98%	21.43%	**5.13%**
Dermatology	52.92%	26.14%	32.43%	**24.26%**	49.80%	62.34%	61.46%	36.00%
Hayes Roth	**37.33%**	59.70%	50.15%	44.65%	59.39%	64.36%	63.46%	38.31%
Heart	39.44%	38.53%	39.27%	30.67%	30.85%	31.20%	28.44%	**16.07%**
HeartAttack	46.67%	45.55%	32.34%	32.97%	33.93%	29.00%	40.48%	**19.20%**
HouseVotes	7.13%	7.48%	10.89%	**3.13%**	8.39%	6.13%	11.99%	3.65%
Ionosphere	15.29%	16.64%	19.67%	12.57%	15.03%	16.22%	19.83%	**12.17%**
Liverdisorder	42.59%	41.53%	30.67%	32.21%	34.21%	30.84%	36.97%	**29.29%**
Lymography	35.43%	29.26%	33.70%	**24.07%**	25.50%	25.50%	29.33%	24.36%
Mammographic	**17.24%**	46.25%	22.85%	19.83%	21.15%	21.38%	30.41%	17.79%
Parkinsons	27.58%	24.06%	18.56%	21.32%	19.32%	**17.41%**	33.81%	17.53%
Pima	35.59%	34.85%	34.51%	32.63%	35.52%	25.78%	27.83%	**24.02%**
Popfailures	5.24%	**5.18%**	7.05%	6.83%	7.63%	7.04%	7.08%	6.33%
Regions2	36.28%	29.85%	33.23%	33.42%	30.16%	38.29%	39.98%	**26.29%**
Saheart	37.48%	34.04%	34.51%	35.11%	34.87%	32.19%	33.90%	**28.50%**
Segment	68.97%	49.75%	66.72%	**32.04%**	51.70%	59.68%	54.25%	45.00%
Sonar	25.85%	30.33%	34.10%	**20.50%**	27.15%	27.90%	37.13%	22.00%
Spiral	47.99%	48.90%	50.22%	45.64%	50.57%	44.87%	50.02%	**13.26%**
Statheart	39.65%	44.04%	44.36%	30.22%	31.41%	31.36%	42.94%	**19.67%**
Student	7.14%	**5.13%**	10.20%	6.93%	5.83%	5.49%	33.26%	5.23%
Transfusion	25.84%	25.68%	**24.87%**	25.92%	25.41%	26.41%	25.67%	26.04%
Wdbc	29.91%	35.35%	12.88%	9.43%	9.52%	7.27%	8.82%	**5.54%**
Wine	59.71%	29.40%	25.43%	27.18%	21.77%	31.41%	31.47%	**9.47%**
Z_F_S	39.37%	47.81%	38.41%	9.27%	17.63%	13.16%	23.37%	**3.73%**
Z_O_N_F_S	65.67%	78.79%	77.08%	67.80%	54.08%	48.70%	68.40%	**41.00%**
ZO_NF_S	43.04%	47.43%	43.75%	8.50%	20.02%	9.02%	22.18%	**4.24%**
ZONF_S	15.62%	11.99%	5.44%	2.52%	3.10%	4.03%	17.41%	**1.98%**
ZOO	10.70%	14.13%	20.27%	16.20%	14.70%	21.93%	33.50%	**9.80%**
AVERAGE	**33.50%**	**33.73%**	**32.77%**	**25.52%**	**27.21%**	**28.54%**	**34.89%**	**19.45%**

Table 4 compares the performance of eight methods on thirty-three classification datasets. The mean percentage error clearly shows that the proposed method is best overall at 19.45%, followed by DNN at 25.52%, then BAYES 27.21%, RBF-KMEANS 28.54%, NEAT 32.77%, BFGS 33.50%, ADAM 33.73%, and GENRBF 34.89%. Relative to the strongest competitor, DNN, the proposed method lowers the average error by 6.07 points, about 24%. The reduction versus the classical BFGS and ADAM is about 14 points, roughly 42%, and versus RBF-KMEANS about 9.1 points, roughly 32%.

At the level of individual datasets, the proposed method delivers strikingly low errors in several cases. On Spiral, it drops to 13.26% while others are around 45–50%; on Wine it reaches 9.47% versus 21–60%; on Wdbc it achieves 5.54% versus 7–35%. On Z_F_S, ZO_NF_S, ZONF_S, and Cleveland it attains the best or tied-best results. On Heart, HeartAttack, Statheart, Regions2, Saheart, Pima, Australian, Alcohol, and HouseVotes, the results are also highly competitive, usually best or within the top two. There are, however, datasets where other methods prevail: DNN clearly leads on Segment and HouseVotes and is very strong on Dermatology; RBF-KMEANS is best on Appendicitis; and ADAM wins narrowly on Student and Balance. In cases like Balance, Popfailures, Dermatology, and Segment, the proposed method is not the top performer, though it remains competitive.

Overall, the proposed method offers the lowest average error and a consistent advantage across a wide range of problems, with substantial gains over all classical and modern baselines. Despite local exceptions where DNN, ADAM, or RBF-KMEANS come out ahead, the approach appears more generalizable and stable, achieving systematically low errors

and large improvements on challenging datasets, which supports its practical use as a default choice for classification.

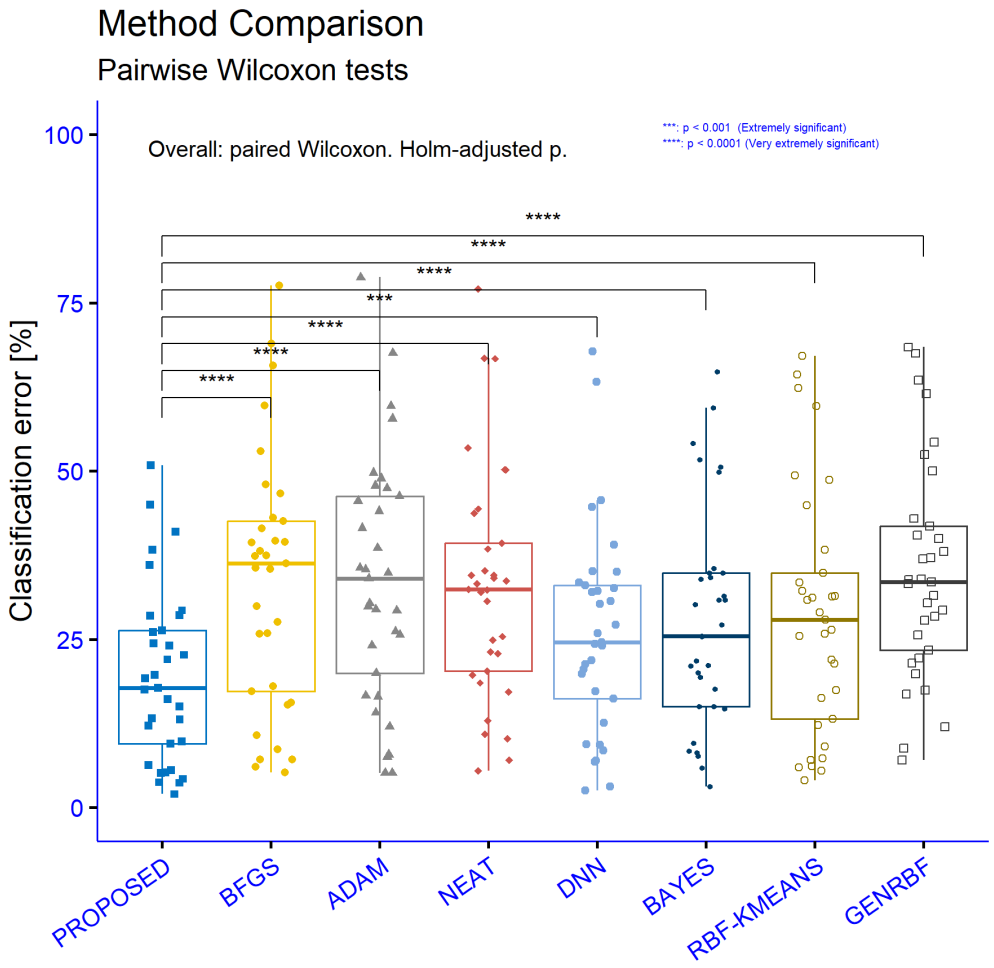

Figure 6. Statistical tests on the experimental results on the classification datasets, using the machine learning methods mentioned in the current article.

Table 5. Pairwise Wilcoxon Results: Proposed vs Baselines on classification datasets (95% CI & Effect Size)

Comparison	n	V	$r_{rank,biserial}$	$conf_{low}$	$conf_{high}$	p	p_{adj}	p_{signif}
PROPOSED vs BFGS	33	26	-9536541889483070	-18505038146370400	-7964957107005870	5667497792.56	17002493377.69	****
PROPOSED vs ADAM	33	32	-9429590017825310	-19420020785066500	-8034989453223090	9370072082.50	187401441650.00	****
PROPOSED vs NEAT	33	11	-9803921568627450	-1840001089212400	-7149974070777490	15363692460.10	9218215476.06	****
PROPOSED vs DNN	33	66	-8823529411764710	-8925029424575650	-3094966481225850	1314556247229.44	1314556247229.44	***
PROPOSED vs BAYES	32	17	-9678030303030300	-10224975985015600	-49000094426908900	4040490189.97	161619607598.76	****
PROPOSED vs RBF-KMEANS	33	13	-9768270944741530	-12110046788330500	-4899979230676410	18357733429.22	9218215476.06	****
PROPOSED vs GENRBF	33	1	-9982174688057040	-1912007131513760	-12009996230590000	619219811.57	4334538680.97	****

The Figure 6 and the Table 5 summarizes paired Wilcoxon signed-rank tests comparing the PROPOSED method against each competitor on the same 33 classification datasets. The column n is the number of paired datasets, V is the Wilcoxon signed-rank statistic, $r_{rank,biserial}$ is the rank-biserial effect size (range -1 to 1, with more negative meaning PROPOSED has lower error), $conf_{low}$ and $conf_{high}$ give the 95% Hodges-Lehmann confidence interval for the median paired difference (PROPOSED - competitor) in percentage point error, p is the raw p-value,p_{adj} is the Holm-adjusted p-value, and p_{signif} is the significance code. Because all confidence intervals are entirely negative, the PROPOSED method consistently shows lower error than each baseline, not just statistical significance but a stable direction of effect across datasets. Adjusted p-values remain very small in every comparison, from 4.33×10^{-6} (vs GENRBF) up to 1.31×10^{-4} (vs DNN), yielding **** everywhere except the DNN comparison, which is ***. Effect sizes are uniformly large in magnitude. The strongest difference is against GENRBF with r≈-0.998 and a 95% CI for the median error reduction of roughly -19.12 to -12.01 percentage points. Very large effects also appear versus NEAT (r≈-0.980, CI ≈ [-18.40, -7.15]) and RBF-KMEANS (r≈−0.977, CI ≈ [-12.11, -4.90]). Comparisons with BFGS (r≈-0.954, CI ≈ [-18.51, -7.96]) and ADAM (r≈-0.943, CI ≈ [-19.42, -8.03]) remain strongly favorable. The smallest, yet still large, effect is against DNN (r≈-0.882) with a clearly negative CI ≈ [-8.93, -3.09]. Taken together, the results show consistent, substantial reductions in classification error for the PROPOSED method across all baselines, with very large effect sizes, tight negative confidence intervals, and significance that survives multiple-comparison correction.

Also, a comparison in terms of precision and recall for the classification datasets, between the original training method for RBF networks and the proposed one is depicted in Table 6.

	RBF - KMEANS		PROPOSED	
DATASET	PRECISION	RECALL	PRECISION	RECALL
Alcohol	0.507	0.639	0.723	0.711
Appendicitis	0.762	0.875	0.804	0.722
Australian	0.604	0.669	0.779	0.756
Balance	0.753	0.741	0.794	0.86
Cleveland	0.268	0.385	0.39	0.392
Circular	0.941	0.948	0.963	0.962
Dermatology	0.305	0.357	0.642	0.589
Hayes Roth	0.34	0.378	0.68	0.632
Heart	0.69	0.688	0.839	0.831
HeartAttack	0.668	0.674	0.779	0.774
HouseVotes	0.938	0.94	0.962	0.966
Ionosphere	0.806	0.847	0.889	0.868
Liverdisorder	0.665	0.673	0.689	0.684
Lymography	0.688	0.742	0.783	0.774
Mammographic	0.793	0.793	0.826	0.826
Parkinsons	0.685	0.8	0.758	0.747
Pima	0.679	0.732	0.744	0.705
Popfailures	0.501	0.93	0.792	0.735
Regions2	0.331	0.502	0.645	0.506
Saheart	0.607	0.641	0.669	0.645
Segment	0.4	0.433	0.603	0.579
Sonar	0.716	0.722	0.805	0.792
Spiral	0.553	0.555	0.868	0.869
Statheart	0.689	0.695	0.797	0.793
Student	0.944	0.955	0.949	0.95
Transfusion	0.533	0.641	0.618	0.534
Wdbc	0.912	0.929	0.952	0.943
Wine	0.676	0.763	0.919	0.907
Z_F_S	0.865	0.871	0.954	0.96
Z_O_N_F_S	0.534	0.52	0.621	0.61
ZO_NF_S	0.9	0.9	0.956	0.6
ZONF_S	0.926	0.947	0.966	0.976
ZOO	0.804	0.809	0.875	0.878
AVERAGE	**0.667**	**0.718**	**0.789**	**0.76**

Table 6. Precision and recall for the original training method for RBFs and the proposed method.

Table 7. Experimental results on the regression datasets, using the machine learning methods mentioned in the this work.

DATASET	BFGS	ADAM	NEAT	DNN	BAYES	RBF-KMEANS	GENRBF	PROPOSED
Abalone	5.69	**4.30**	9.88	6.91	4.81	7.37	9.98	6.12
Airfoil	**0.003**	0.005	0.067	0.004	0.004	0.27	0.121	0.004
Auto	60.97	70.84	56.06	13.26	27.03	17.87	16.78	**8.81**
Baseball	119.63	**77.90**	100.39	110.22	88.76	93.02	98.91	88.05
BK	0.28	0.03	0.15	**0.02**	0.023	0.02	0.023	0.022
BL	2.55	0.28	0.05	0.006	0.46	0.013	0.005	**0.0004**
Concrete	0.066	0.078	0.081	0.021	0.013	0.011	0.015	**0.005**
Dee	2.36	0.630	1.512	0.31	0.28	0.17	0.25	**0.15**
Housing	97.38	80.20	56.49	65.18	57.39	57.68	95.69	**15.36**
Friedman	**1.26**	22.90	19.35	2.75	3.79	7.23	16.24	5.99
FA	0.426	0.11	0.19	0.02	0.051	0.015	0.15	**0.013**
FY	0.22	**0.038**	0.08	0.039	0.21	0.041	0.041	0.054
HO	0.62	0.035	0.169	0.026	0.034	0.03	0.076	**0.009**
Laser	**0.015**	0.03	0.084	0.045	0.026	0.03	0.075	0.016
Mortgage	8.23	9.24	14.11	9.74	3.01	1.45	1.92	**0.23**
PL	0.29	0.117	0.098	0.056	0.056	2.12	0.155	**0.023**
Plastic	20.32	11.71	20.77	3.82	3.66	8.62	25.91	**2.28**
PY	0.578	0.09	0.075	0.028	0.401	**0.012**	0.029	0.021
Quake	0.42	0.06	0.298	0.04	0.093	0.07	0.79	**0.036**
SN	0.40	0.026	0.174	0.032	0.055	0.027	0.027	**0.026**
Stock	302.43	180.89	12.23	39.08	14.43	12.23	25.18	**1.44**
Treasury	9.91	11.16	15.52	13.76	3.74	2.02	1.89	**0.47**
AVERAGE	**28.82**	**21.39**	**13.99**	**11.82**	**9.18**	**9.56**	**13.38**	**5.87**

Table 7 evaluates the performance of eight regression methods on twenty-one datasets using absolute errors. The average error shows a clear overall advantage for the proposed method at 5.87, followed by BAYES at 9.18, RBF-KMEANS at 9.56, DNN at 11.82, NEAT at 13.99, GENRBF at 13.38, ADAM at 21.39, and BFGS at 28.82. Relative to the best competing average, BAYES, the proposed method reduces error by about 3.31 points (≈36%). The reduction versus RBF-KMEANS is about 3.69 points (≈39%), versus DNN about 5.95 points (≈50%), and relative to NEAT and GENRBF the drops are roughly 58% and 56%, respectively. The advantage is even larger against ADAM and BFGS, where the mean error is nearly halved or more.

Across individual datasets, the proposed method attains the best value in roughly two thirds of the cases. It is clearly first on Auto, BL, Concrete, Dee, Housing, FA, HO, Mortgage, PL, Plastic, Quake, Stock, and Treasury, with particularly large margins on Housing and Stock where errors fall to 15.36 and 1.44 while other methods range from tens to hundreds. On Airfoil it is essentially tied with the best at 0.004, while BFGS is slightly lower at 0.003. There are datasets where other methods lead, such as Abalone where ADAM and BAYES are ahead; Friedman and Laser where BFGS gives the best value; BK where DNN and RBF-KMEANS lead; and PY where RBF-KMEANS is lower. Despite these isolated exceptions, the proposed method remains consistently among the top performers and most often the best.

Overall, the proposed approach combines a very low average error with broad superiority across diverse problem types and error scales, from thousandths to very large magnitudes. The consistency of the gains and the size of the margins over all baselines indicate it is the most efficient and generalizable choice among the regression methods considered.

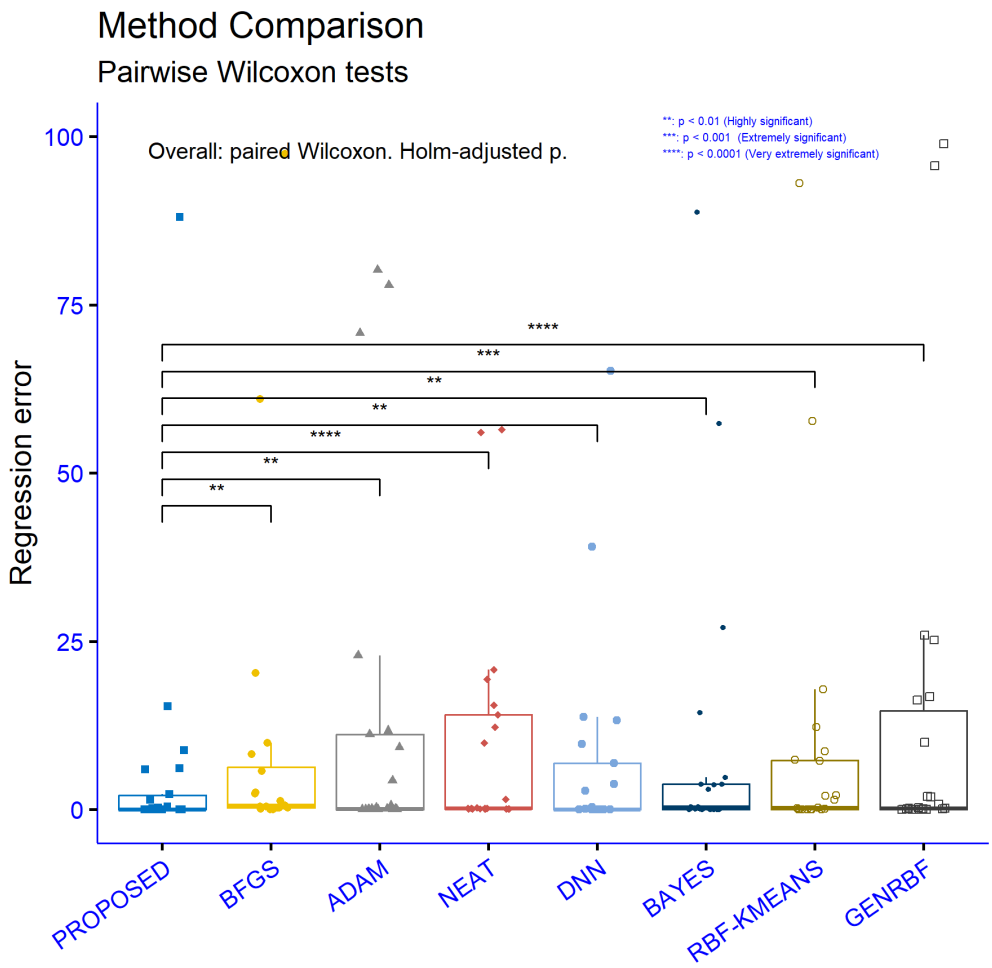

Figure 7. Statistical comparison on the results obtained from the application of the set of machine learning methods to the regression datasets.

Table 8. Pairwise Wilcoxon Results: Proposed vs Baselines on regression datasets (95% CI & Effect Size)

Comparison	n	V	$r_{rank,biserial}$	$conf_{low}$	$conf_{high}$	p	p_{adj}	p_{signif}
PROPOSED vs BFGS	22.00	28.00	-8893280632411070	-20509998654975500	-31607290631815900.0	14644689859755000.00	5541059559701440.00	**
PROPOSED vs ADAM	21.00	33.00	-8571428571428570	-2734490876933390	-41569559046521200.0	4370173456024450.00	626554825580216O.00	**
PROPOSED vs NEAT	22.00	0.00	-1	-9925969319615250	-16000237429855300.0	43012521191.44	3010876483400.92	***
PROPOSED vs DNN	21.00	23.00	-9004329004329000	-11086980158184000	-12031295944894700.0	13852648899253600.00	5541059559701440.00	**
PROPOSED vs BAYES	21.00	30.00	-8701298701298700	-5839992445666000	-33511395308536000.0	31327741279010800.00	626554825580216O.00	**
PROPOSED vs RBF-KMEANS	22.00	14.00	-9446640316205530	-4523464832268890	-3407835793499720.0	276740186614.05	13837009330702700.00	**
PROPOSED vs GENRBF	22.00	6.00	-9762845849802370	-10249984407899800	-9806889388206810.0	9774004411.05	586440264662.75	***

The Figure 7 and the Table 8 summarizes paired Wilcoxon signed-rank tests between PROPOSED and each method on the same regression datasets. In every comparison the 95% confidence interval is entirely negative, so PROPOSED consistently attains lower error than each baseline. Holm-adjusted p-values range from about 5.86×10^{-4} to 0.0063, yielding ** or *** across all pairings, indicating strong though not extreme significance. Effect sizes are very large in absolute value, implying a consistent sign of the differences across datasets. The strongest dominance is against NEAT with r≈-1 and V=0, meaning that in every non-tied pair PROPOSED was better, with a confidence interval roughly [-9.93, -0.16]. Similarly large effects appear against GENRBF (r≈-0.976, CI [-10.25, -0.098]) and RBF-KMEANS (r≈-0.945, CI [-4.52, -0.034]), the upper bound near zero indicates that the typical improvement can range from very small to several points depending on the dataset. Against BFGS and ADAM the effects remain very large (r≈-0.889 and r≈-0.857, respectively) with wider intervals [-20.51, -0.316] and [-27.34, -0.0416], showing substantial heterogeneity in the magnitude of error reduction while the direction remains in favor of PROPOSED. The most challenging comparison is with DNN: although |r| is still very large (≈0.900), the CI is narrow and close to zero [-11.09, -0.012], implying that while superiority is consistent, the typical error reduction may be small in many cases.

Overall, the results demonstrate that PROPOSED systematically outperforms all alternatives on regression, with very large rank-based effect sizes, negative and robust confidence intervals, and significance that survives multiple-comparison correction. The strength of the improvement varies by problem and is more modest against DNN, but the direction of the effect is consistently in favor of PROPOSED across all comparisons.

3.2. Experiments with different values of scale factor F

In order to evaluate the stability and reliability of the current work when its critical parameters are altered, a series of additional experiments were executed. In one of them, the stability of the technique was studied with the change of the scale factor F. This factor controls the width of the value interval for the network parameters and is a multiple of the initial values estimated by the K-means method of the first phase. In this series of experiments, the value of F was altered in the range $[1, 8]$.

Table 9 presents the effect of the scale factor F on the performance of the proposed machine learning model. The parameter F takes four different values, 1, 2, 4, and 8, and for each dataset the classification error rate is reported. Analyzing the mean values, it is observed that $F = 2$ and $F = 4$ achieve the best overall performance, with average errors of 19.45% and 18.53% respectively, compared to 20.99% for $F = 1$ and 18.60% for $F = 8$. This indicates that selecting an intermediate value of the initialization factor improves performance, reducing the error by about two percentage points relative to the baseline case of $F = 1$. At the individual dataset level, interesting patterns emerge. For example, in Sonar the error drops significantly from 32.90% at $F = 1$ to 18.75% at $F = 4$, suggesting that the parameter F strongly influences performance in certain problems. In contrast, in Spiral increasing F worsens the results, as the error rises from 12.03% at $F = 1$ to 23.56% at $F = 8$. Similarly, in the Australian dataset a gradual increase of F from 1 to 8 systematically improves performance, reducing the error from 24.04% to 20.59%. Overall, the data show that the effect of the scale factor is not uniform across all problems, but the general trend indicates improvement when F increases from 1 to 2 or 4. Choosing $F = 4$ appears to yield the best mean result, although the difference compared with $F = 8$ is very small. Therefore, it can be concluded that tuning this parameter plays an important role in the stability and accuracy of the model, and that intermediate values such as 4 constitute a good general choice.

Table 9. Results from the application of the proposed method to the classification datasets, where the critical parameter F was altered from $F = 1$ to $F = 8$.

DATASET	$F = 1$	$F = 2$	$F = 4$	$F = 8$
Alcohol	28.83%	**28.57%**	28.83%	30.09%
Appendicitis	14.60%	15.00%	**14.40%**	15.50%
Australian	24.04%	22.67%	21.52%	**20.59%**
Balance	21.03%	13.11%	11.87%	**11.44%**
Cleveland	**50.45%**	50.86%	51.59%	50.90%
Circular	4.13%	5.13%	3.67%	**3.49%**
Dermatology	38.34%	36.00%	35.83%	**34.97%**
Hayes Roth	51.85%	38.31%	**32.62%**	33.92%
Heart	17.26%	16.07%	15.63%	**15.30%**
HeartAttack	22.07%	19.20%	19.30%	**19.07%**
HouseVotes	4.13%	3.65%	**3.39%**	4.81%
Ionosphere	14.69%	12.17%	8.83%	**7.51%**
Liverdisorder	29.35%	29.29%	**28.53%**	29.23%
Lymography	26.86%	24.36%	**18.07%**	19.86%
Mammographic	18.21%	17.79%	**16.75%**	17.05%
Parkinsons	18.32%	17.53%	15.68%	**14.05%**
Pima	23.53%	24.02%	23.72%	**23.26%**
Popfailures	7.83%	6.33%	5.15%	**4.69%**
Regions2	26.27%	26.29%	26.15%	**25.73%**
Saheart	29.24%	**28.50%**	28.74%	29.41%
Segment	45.08%	45.00%	42.14%	**42.10%**
Sonar	32.90%	22.00%	18.75%	**18.05%**
Spiral	**12.03%**	13.26%	16.66%	23.56%
Statheart	**19.30%**	19.67%	20.00%	19.44%
Student	6.33%	5.23%	**5.10%**	5.55%
Transfusion	25.54%	26.04%	25.66%	**24.42%**
Wdbc	**4.86%**	5.54%	5.75%	5.29%
Wine	12.18%	9.47%	8.59%	**7.65%**
Z_F_S	4.37%	3.73%	3.73%	**3.37%**
Z_O_N_F_S	**39.80%**	41.00%	40.04%	40.80%
ZO_NF_S	4.26%	4.24%	4.58%	**3.78%**
ZONF_S	2.52%	1.98%	2.58%	**1.96%**
ZOO	12.40%	9.80%	7.60%	**6.90%**
AVERAGE	**20.99%**	**19.45%**	**18.53%**	**18.60%**

Table 10 shows the effect of the scale factor F on the performance of the proposed regression model. Based on the mean errors, the best overall performance occurs at $F = 4$ with an average error of 5.68, while the values for $F = 1$, $F = 2$ and $F = 8$ are 5.94, 5.87, and 5.78, respectively. The differences across the four settings are not large, but they indicate that intermediate values and especially $F = 4$ tend to offer the best accuracy stability trade-off. At the level of individual datasets, substantial variations are observed. For Friedman the reduction is dramatic, with error dropping from 6.74 at $F = 1$ to 1.41 at $F = 8$, highlighting that proper tuning of F can have a strong impact on performance. Laser shows a similarly large improvement, from 0.027 at $F = 1$ to just 0.0024 at $F = 8$. Mortgage also improves markedly, from 0.67 at $F = 1$ to 0.015 at $F = 8$. By contrast, in some datasets the value of F has little practical effect, such as Quake and HO, where errors remain nearly constant regardless of F. There are also cases like Housing where increasing F degrades performance, with error rising from 14.64 at $F = 1$ to 18.48 at $F = 8$. Overall, the results indicate that the scale factor F has a significant but nonuniform influence on model performance. In some datasets it sharply reduces error, while in others its impact is negligible or even negative. Nevertheless, the aggregate picture based on the mean errors

suggests that $F = 4$ and $F = 8$ yield the most reliable results, with $F = 4$ being the preferred choice for a general-purpose setting.

Table 10. Experimental results on the regression datasets where the proposed method was applied and the critical parameter F was altered from $F = 1$ to $F = 8$.

DATASET	$F = 1$	$F = 2$	$F = 4$	$F = 8$
Abalone	6.70	6.12	5.70	**5.56**
Airfoil	**0.004**	0.004	0.004	0.004
Auto	10.04	**8.81**	9.82	10.92
Baseball	87.01	88.05	**85.87**	86.76
BK	0.023	0.022	0.024	**0.02**
BL	0.01	0.0004	0.0002	**0.00007**
Concrete	0.008	**0.005**	0.005	0.006
Dee	**0.15**	0.15	0.16	0.16
Housing	**14.64**	15.36	17.34	18.48
Friedman	6.74	5.99	2.06	**1.41**
FA	**0.012**	0.013	0.012	0.013
FY	0.055	0.054	0.054	**0.053**
HO	**0.009**	0.009	0.01	0.009
Laser	0.027	0.016	0.005	**0.0024**
Mortgage	0.67	0.23	0.035	**0.015**
PL	0.023	0.023	0.023	**0.022**
Plastic	2.32	2.28	2.26	**2.22**
PY	0.019	0.021	0.013	**0.011**
Quake	**0.036**	0.036	0.036	0.036
SN	**0.024**	0.026	0.025	0.024
Stock	1.69	**1.44**	1.49	1.48
Treasury	0.57	0.47	0.035	**0.031**
AVERAGE	**5.94**	**5.87**	**5.68**	**5.78**

In Figure 8, the significance levels are presented for the comparisons between different values of the parameter F in the proposed machine learning method based on the classification datasets. The analysis shows that the comparison between $F = 1$ and $F = 2$ results in high statistical significance with $p < 0.01$, indicating that the transition from the initial value to $F = 2$ has a substantial impact on performance. Similarly, the comparison between $F = 2$ and $F = 4$ also shows high statistical significance with $p < 0.01$, suggesting that further increasing the parameter continues to positively affect the results. However, the comparison between $F = 4$ and $F = 8$ is characterized as not statistically significant, since $p > 0.05$, which means that increasing the parameter beyond $F = 4$ does not bring a meaningful difference in performance. Overall, the findings indicate that smaller values of F play a critical role in improving the model, while increases beyond 4 do not lead to further statistically significant improvements.

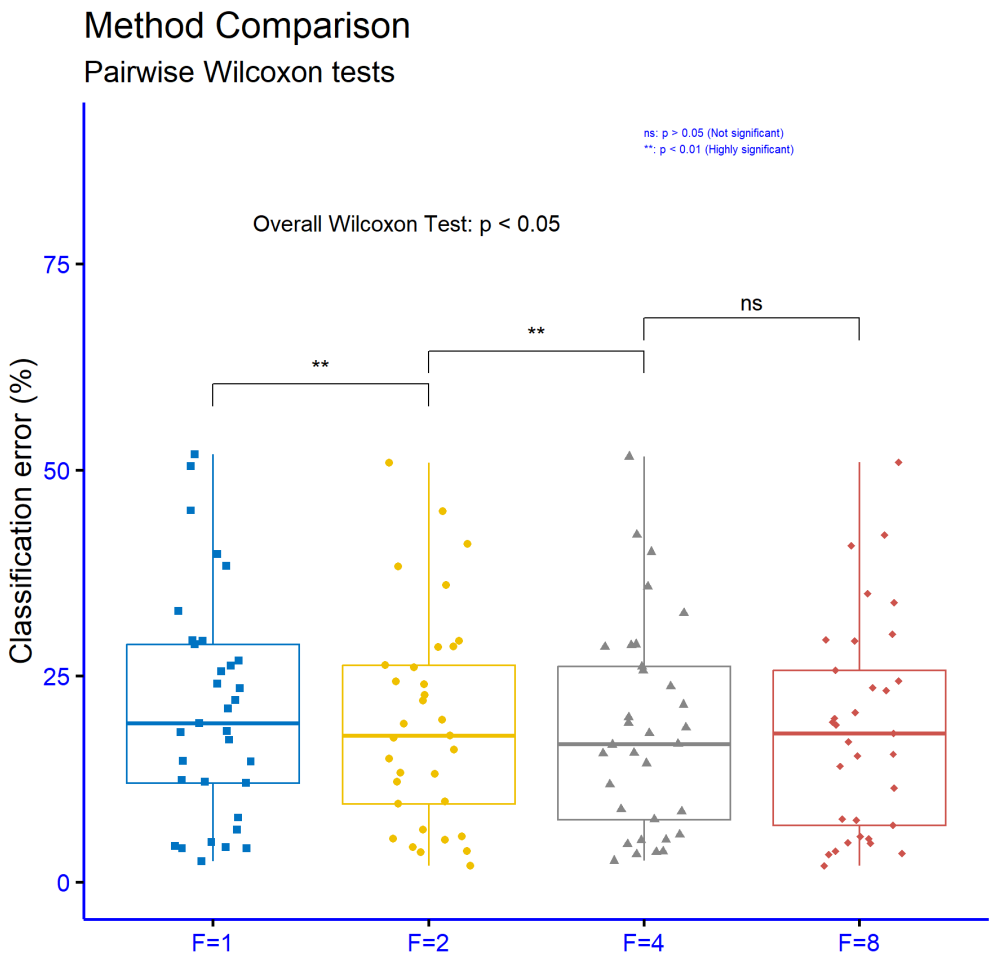

Figure 8. Statistical tests on the results obtained by the application of the suggested work to the classification datasets, where the parameter F was altered.

In Figure 9, the significance levels are presented for the comparisons between different values of the parameter F in the proposed method based on the regression datasets. The results show that none of the comparisons $F = 1$ vs $F = 2$, $F = 2$ vs $F = 4$, and $F = 4$ vs $F = 8$ exhibit statistically significant differences, since in all cases $p > 0.05$. This means that variations in the parameter F do not substantially affect the performance of the model in regression problems. Therefore, it can be concluded that the choice of the F value is not of critical importance for these datasets and that the model remains stable regardless of the specific setting of this parameter.

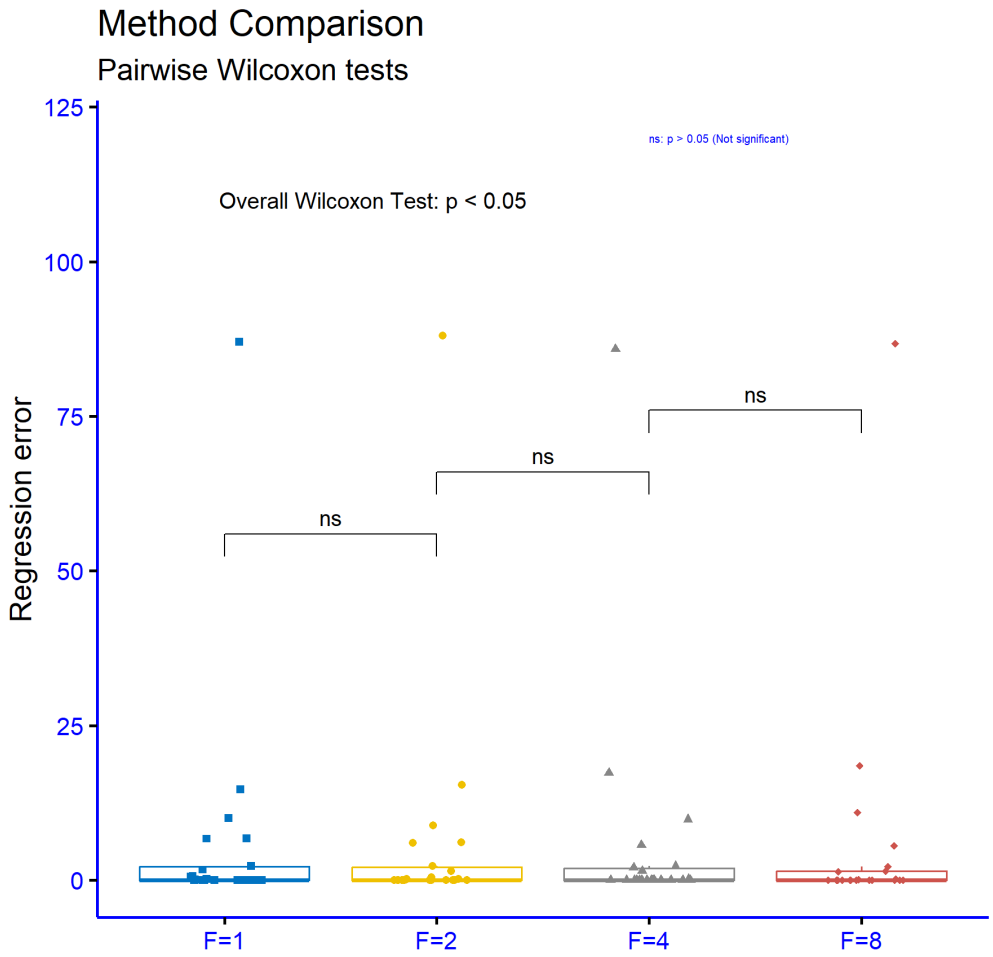

Figure 9. Statistical tests on the results obtained by the application of the proposed method to the regression datasets, where the scale parameter F was altered.

3.3. Experiments with differential initialization methods for variances

Additionally, the stability of the proposed method was checked using a different way of calculating the range of values of the σ parameters of the radial functions. In this work, the value of the variance produced by the K-means algorithm was used as an initial estimate of the σ parameters. This calculation scheme is denoted as σ_1 in the following experimental tables. In this additional set of experiments, two more techniques were used, which will be denoted as σ_{avg} and σ_{max} in the following tables. In the σ_{avg} the following calculation is performed:

$$\sigma_{\text{avg}} = \frac{1}{k}\sum_{i=1}^{k}\sigma_i \tag{11}$$

Subsequently σ_{avg} is used to determine the range of values of the σ parameters of the radial functions of the network. In the σ_{avg} the following quantity is calculated:

$$\sigma_{\text{max}} = \max \sigma_i \tag{12}$$

Afterwards this quantity is used for the determination of the range of values for the σ parameters of the radial functions.

Table 11 presents the effect of three different calculation techniques for the σ parameters used in the radial basis functions of the RBF model. The techniques are a fixed value σ_1, the mean distance-based initialization $\left(\sigma_{\text{avg}}\right)$, and the maximum distance–based ini-

tialization ($\sigma_{\max}$). Based on the mean errors, the maximum-distance technique yields the lowest overall error at 19.18%. Very close is the mean-distance technique at 19.27%, while the simple σ_1 initialization has a slightly higher error of 19.45%. Although the differences among the three approaches are small, the two adaptive methods $\left(\sigma_{\text{avg}} \text{ and } \sigma_{\max}\right)$ tend to produce marginally better overall performance. At the individual dataset level, behaviors vary. For example, on Wine the $\sigma_{\max}$ choice reduces error to 7.06%, far below the 9.47% obtained with σ_1. On Dermatology, σ_1 performs better than the other two, whereas on Segment the mean-based option is preferable. In some cases the differences are minor e.g., Circular, Pima, and Popfailures where all techniques are comparable; in others the choice of technique materially affects performance, as in Transfusion, where error drops from 26.04% with σ_1 to about 22.78% with the other two methods. Overall, the statistical picture indicates that no single technique dominates across all datasets. Nevertheless, methods that adapt σ to the geometry of the data $\left(\sigma_{\text{avg}} \text{ and } \sigma_{\max}\right)$ tend to yield more reliable and stable results, while the fixed value lags slightly. The average differences are modest, but for certain problems the choice can significantly impact final performance.

Table 11. Experiments on the classification datasets using the proposed method and different mechanisms for the calculation of the quantities σ used in the radial functions.

DATASET	σ_1	σ_{avg}	$\sigma_{\max}$
Alcohol	28.57%	28.47%	**26.17%**
Appendicitis	15.00%	**14.20%**	15.70%
Australian	**22.67%**	25.14%	29.96%
Balance	13.11%	12.92%	**12.23%**
Cleveland	**50.86%**	51.76%	51.24%
Circular	5.13%	4.78%	**4.45%**
Dermatology	**36.00%**	37.54%	37.09%
Hayes Roth	38.31%	38.00%	**35.69%**
Heart	16.07%	16.52%	**15.41%**
HeartAttack	19.20%	19.70%	**18.97%**
HouseVotes	3.65%	3.31%	**3.22%**
Ionosphere	**12.17%**	13.00%	12.83%
Liverdisorder	29.29%	28.38%	**27.77%**
Lymography	24.36%	**22.43%**	23.50%
Mammographic	17.79%	**17.28%**	17.41%
Parkinsons	17.53%	**14.74%**	14.89%
Pima	24.02%	**23.28%**	23.91%
Popfailures	6.33%	6.37%	**6.24%**
Regions2	26.29%	**25.47%**	25.61%
Saheart	28.50%	28.89%	**28.28%**
Segment	45.00%	**43.65%**	46.36%
Sonar	22.00%	21.90%	**21.30%**
Spiral	**13.26%**	13.73%	13.37%
Statheart	19.67%	20.15%	**19.00%**
Student	**5.23%**	5.58%	5.23%
Transfusion	26.04%	**22.78%**	22.79%
Wdbc	5.54%	5.22%	**5.21%**
Wine	9.47%	7.93%	7.06%
Z_F_S	3.73%	**3.70%**	3.73%
Z_O_N_F_S	41.00%	**40.20%**	41.12%
ZO_NF_S	**4.24%**	4.42%	4.84%
ZONF_S	1.98%	**1.92%**	2.06%
ZOO	**9.80%**	12.50%	10.30%
AVERAGE	**19.45%**	**19.27%**	**19.18%**

Table 12 presents the effect of three different calculation techniques for the σ parameters used in the radial basis functions of RBF model. Based on the mean errors, the distance–average method yields the lowest overall error at 5.81. Very close is the fixed value σ_1 with a mean error of 5.87, while the maximum-distance method shows a slightly higher mean error of 5.96. The difference among the three methods is small, indicating that all can deliver comparable performance at a general level, with a slight advantage for the distance–average approach. At the level of individual datasets, however, significant variations are observed. For example, in Mortgage the $\sigma_{\max}$ method reduces the error dramatically from 0.23 with σ_1 to 0.021, while σ_{avg} also provides a much better result with 0.041. In Treasury the improvement is again substantial, as the error decreases from 0.47 with σ_1 to just 0.08 using $\sigma_{\max}$. In Stock the reduction is clear, from 1.44 to 1.23, while in Plastic both σ_{avg} and $\sigma_{\max}$ yield lower errors than σ_1. On the other hand, in datasets such as Housing, the use of $\sigma_{\max}$ worsens performance, increasing the error from 15.36 with σ_1 to 19.45. Similarly, in Auto and Baseball the lowest errors are obtained with σ_1, whereas the alternative techniques result in slightly worse performance. Overall, the results show that the choice of calculation technique for σ can significantly affect performance in certain problems, while in others the difference is negligible. Although no method consistently outperforms the others across all datasets, the distance–average method appears slightly more reliable overall, while the maximum-distance method can in some cases produce very large improvements but in others lead to a degradation of performance.

Table 12. Results obtained by the application of the proposed method on the regression datasets with different calculation mechanisms for the σ used in the radial functions.

DATASET	σ_1	σ_{avg}	$\sigma_{\max}$
Abalone	6.12	6.06	5.43
Airfoil	0.004	0.003	0.003
Auto	8.81	9.80	10.44
Baseball	88.05	86.13	85.89
BK	0.022	0.022	0.022
BL	0.0004	0.008	0.0004
Concrete	0.005	0.005	0.005
Dee	0.15	0.16	0.16
Housing	15.36	15.57	19.45
Friedman	5.99	6.21	6.02
FA	0.013	0.012	0.012
FY	0.054	0.055	0.055
HO	0.009	0.009	0.01
Laser	0.016	0.018	0.011
Mortgage	0.23	0.041	0.021
PL	0.023	0.022	0.022
Plastic	2.28	2.21	2.19
PY	0.021	0.02	0.022
Quake	0.036	0.036	0.036
SN	0.026	0.026	0.025
Stock	1.44	1.32	1.23
Treasury	0.47	0.15	0.08
AVERAGE	**5.87**	**5.81**	**5.96**

In Figure 10, the significance levels are presented for the comparisons of different computation techniques for the σ parameters in the radial basis functions of the proposed machine learning model, based on the classification datasets. The comparisons performed σ_1 vs σ_{avg}, σ_1 vs $\sigma_{\max}$, and σ_{avg} vs $\sigma_{\max}$ did not show any statistically significant differences, since in all cases $p > 0.05$. This indicates that the choice of compuation

method for the σ parameters does not substantially affect the performance of the model on classification problems. Therefore, it can be concluded that the model maintains stable performance regardless of which of the three computation techniques is used.

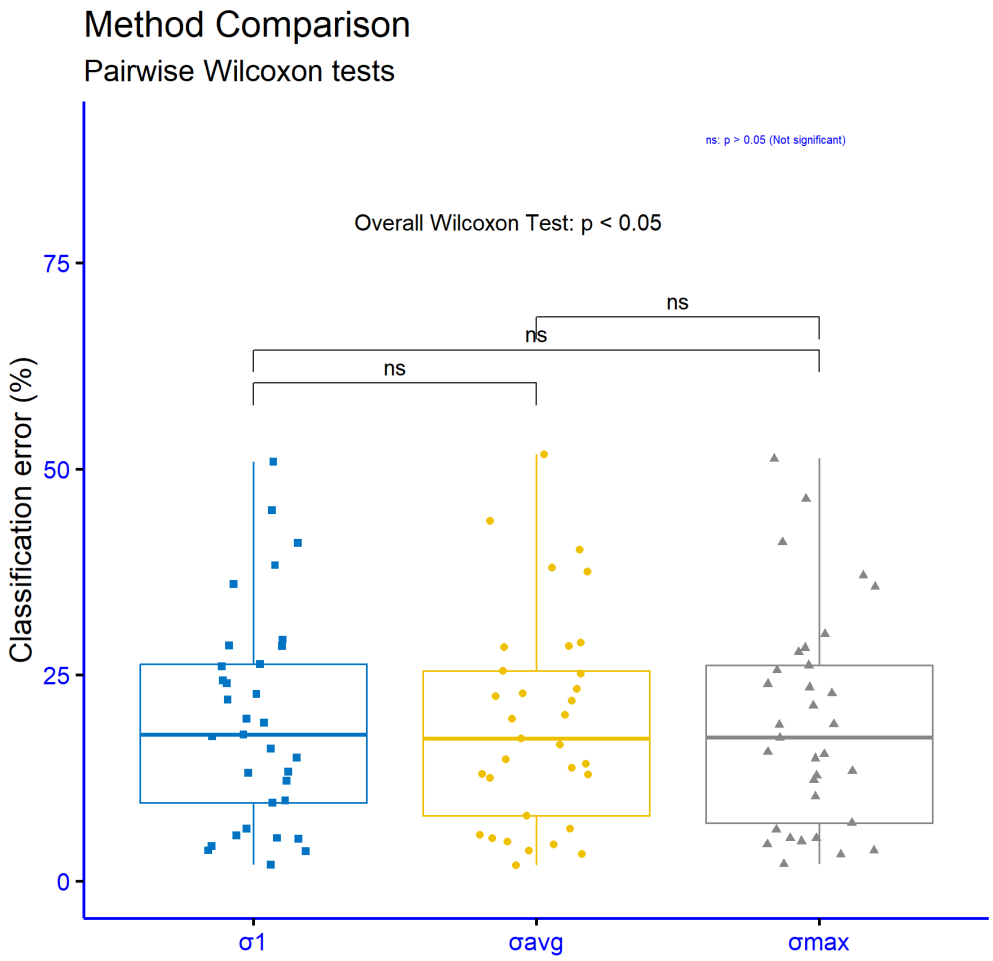

Figure 10. Statistical tests for the results obtained by the application of the current method on the classification datasets using a series of computation techniques for the range of σ values in the radial functions.

In Figure 11, the significance levels are presented for the comparisons of different computation techniques for the σ parameters in the radial basis functions of the proposed machine learning model, based on the regression datasets. The comparisons examined σ_1 vs σ_{avg}, σ_1 vs σ_{max}, and σ_{avg} vs σ_{max} did not show any statistically significant differences, since in all cases $p > 0.05$. This means that the choice of computation method for the σ parameters does not have a substantial impact on the performance of the model in regression problems. Therefore, it can be concluded that the model demonstrates stable and consistent behavior regardless of which initialization technique is applied.

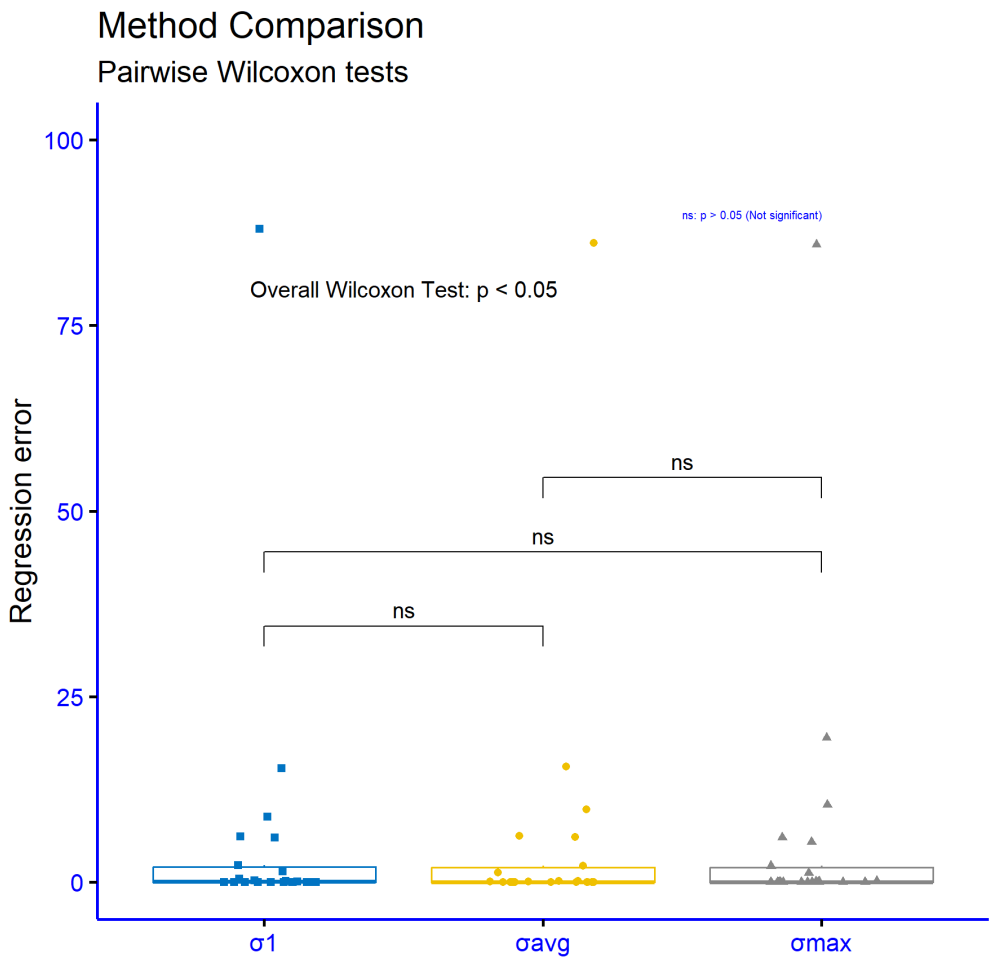

Figure 11. Statistical tests on the produced results from the application of the current work to the regression datasets, using a series of computation techniques for the range of σ values in the radial functions.

3.4. Experiments with the number of generations N_g

An additional experiments was executed, where the number of generations was altered from $N_g = 50$ to $N_g = 400$. Table 13 presents the effect of the number of generations (N_g) on the performance of the proposed model. The overall trend is downward: the mean error decreases from 20.56% at 50 generations to 19.46% at 100, essentially stabilizes at 200 with 19.45%, and improves slightly further at 400 to 19.11%. Thus, the largest gain arrives early, from 50 to 100 generations (about 1.1 points), after which returns diminish, with small but tangible additional gains. At the dataset level the behavior varies. There are cases with clear improvements as N_g increases, such as Alcohol (34.11%→27.02%), Australian (25.23%→21.39%), Ionosphere (13.94%→11.17%), Spiral (16.66%→12.45%), and Z_O_N_F_S (45.14%→38.26%), where more generations yield substantial benefits. In other problems the best value occurs around 100–200 generations and then plateaus or slightly worsens, as in Wdbc (best 4.84% at 100), Student (4.85% at 100), Lymography (21.64% at 100), ZOO (8.70% at 100), ZONF_S (1.98% at 200), and Z_F_S (3.73% at 200). A few datasets show mild degradation with higher N_g, such as Wine (7.59%→10.24%), Parkinsons (17.32%→17.63%), and to a lesser extent Saheart, indicating that beyond a point further search is not beneficial for all problems. Overall, 100 generations deliver the major error reduction and represent an efficient "sweet spot," while 200–400 generations extract modest additional gains and, in

some datasets, meaningful improvements, at the cost of more computation and occasional local regressions.

Table 13. Experimental results using the proposed method on the classification datasets where the number of generations N_g was altered from $N_g = 50$ to $N_g = 400$.

DATASET	$N_g = 50$	$N_g = 100$	$N_g = 200$	$N_g = 400$
Alcohol	34.11%	31.32%	28.57%	**27.02%**
Appendicitis	14.90%	**14.30%**	15.00%	14.90%
Australian	25.23%	24.96%	22.67%	**21.39%**
Balance	14.98%	14.11%	**13.11%**	13.52%
Cleveland	52.00%	51.31%	**50.86%**	51.38%
Circular	**3.75%**	3.82%	5.13%	3.82%
Dermatology	47.86%	36.29%	**36.00%**	36.46%
Hayes Roth	40.54%	**36.77%**	38.31%	36.77%
Heart	16.19%	16.37%	**16.07%**	16.26%
HeartAttack	21.30%	21.63%	**19.20%**	20.07%
HouseVotes	4.09%	3.65%	3.65%	**3.61%**
Ionosphere	13.94%	12.57%	12.17%	**11.17%**
Liverdisorder	**29.06%**	29.23%	29.29%	29.06%
Lymography	22.14%	**21.64%**	24.36%	21.86%
Mammographic	**17.19%**	17.25%	17.79%	17.78%
Parkinsons	17.32%	**17.11%**	17.53%	17.63%
Pima	24.07%	24.38%	**24.02%**	24.28%
Popfailures	6.63%	**5.92%**	6.33%	6.15%
Regions2	**26.02%**	26.14%	26.29%	26.13%
Saheart	**28.28%**	28.63%	28.50%	29.61%
Segment	43.28%	42.70%	45.00%	**41.35%**
Sonar	22.65%	**21.20%**	22.00%	22.20%
Spiral	16.66%	14.47%	13.26%	**12.45%**
Statheart	20.22%	20.67%	19.67%	**19.63%**
Student	4.98%	**4.85%**	5.23%	5.45%
Transfusion	25.47%	**25.32%**	26.04%	25.84%
Wdbc	5.14%	**4.84%**	5.54%	5.39%
Wine	**7.59%**	8.53%	9.47%	10.24%
Z_F_S	4.13%	4.10%	**3.73%**	4.40%
Z_O_N_F_S	45.14%	43.04%	41.00%	**38.26%**
ZO_NF_S	4.14%	**4.00%**	4.24%	4.02%
ZONF_S	2.30%	2.36%	**1.98%**	2.02%
ZOO	17.10%	**8.70%**	9.80%	10.60%
AVERAGE	**20.56%**	**19.46%**	**19.45%**	**19.11%**

Table 14 examines the effect of the number of generations (N_g) on the performance of the proposed regression model. At the level of average error, the best value appears at 100 generations with 5.61, marginally better than 50 generations at 5.65, while at 200 and 400 generations the mean error increases slightly to 5.87 and 5.86, respectively. This suggests that most of the benefit is achieved early and that further increasing the number of generations does not yield systematic improvement and may even introduce a small deterioration in overall performance.

Across individual datasets the picture is heterogeneous. Clear improvements with more generations are observed in Abalone, where the error steadily drops to 5.88 at 400 generations, in Friedman, with a continuous decline to 5.66, in Stock, improving to 1.33, and in Treasury, where performance stabilizes at 0.47 from 200 generations onward. In other problems the "sweet spot" is around 200 generations: for example, in Mortgage the error falls from 0.66 to 0.23 at 200 before rising again, in Housing it improves to 15.36 at 200

but worsens at 400, in BL the minimum 0.0004 occurs at 200, and in Concrete and HO there is a small but real improvement near 200. There are also cases where more generations seem to burden performance, such as Baseball and BK, where the error rises as N_g increases. In several datasets the number of generations has little practical effect, with near-constant values in Airfoil, Quake, SN, and PL and only minor fluctuations in Dee, FA, Laser, and FY.

Overall, 100 generations provide an efficient and safe choice with the lowest mean error, while 200 generations can deliver the best results on specific datasets at the risk of small regressions elsewhere. Further increasing to 400 generations does not offer a general gain and may lead to slight degradation in some problems, pointing to diminishing returns and possible overfitting or instability depending on the dataset.

Table 14. Experimental results using the proposed method on the regression datasets where the number of generations N_g was altered from $N_g = 50$ to $N_g = 400$.

DATASET	$N_g = 50$	$N_g = 100$	$N_g = 200$	$N_g = 400$
Abalone	6.35	6.11	6.12	**5.88**
Airfoil	**0.004**	0.004	0.004	0.004
Auto	10.27	9.49	**8.81**	9.65
Baseball	**78.73**	79.89	88.05	84.40
BK	**0.021**	0.021	0.022	0.025
BL	0.006	0.003	**0.0004**	0.006
Concrete	0.006	0.006	**0.005**	0.005
Dee	0.15	0.16	**0.15**	0.16
Housing	15.96	15.82	**15.36**	18.53
Friedman	7.41	6.54	5.99	**5.66**
FA	**0.012**	0.013	0.013	0.013
FY	0.055	0.055	**0.054**	0.057
HO	0.01	0.01	**0.009**	0.01
Laser	0.017	**0.015**	0.016	0.015
Mortgage	0.66	0.66	**0.23**	0.48
PL	**0.023**	0.023	0.023	0.023
Plastic	2.28	2.29	**2.28**	2.28
PY	0.02	0.023	0.021	**0.02**
Quake	**0.036**	0.036	0.036	0.036
SN	**0.026**	0.026	0.026	0.026
Stock	1.70	1.57	1.44	**1.33**
Treasury	0.65	0.61	**0.47**	0.47
AVERAGE	**5.65**	**5.61**	**5.87**	**5.86**

Figure 12 shows that increasing the number of generations from $N_g = 50$ to $N_g = 100$ yields a statistically significant improvement ($p < 0.01$, **), indicating a meaningful reduction in error in this range. By contrast, the comparisons $N_g = 100$ vs $N_g = 200$ and $N_g = 200$ vs $N_g = 400$ are not statistically significant ($p > 0.05$, ns), which means that further increasing generations beyond 100 does not produce a consistent additional gain in performance. Overall, the results suggest that the main benefit is achieved early up to about 100 generations after which returns diminish and the differences are not statistically meaningful.

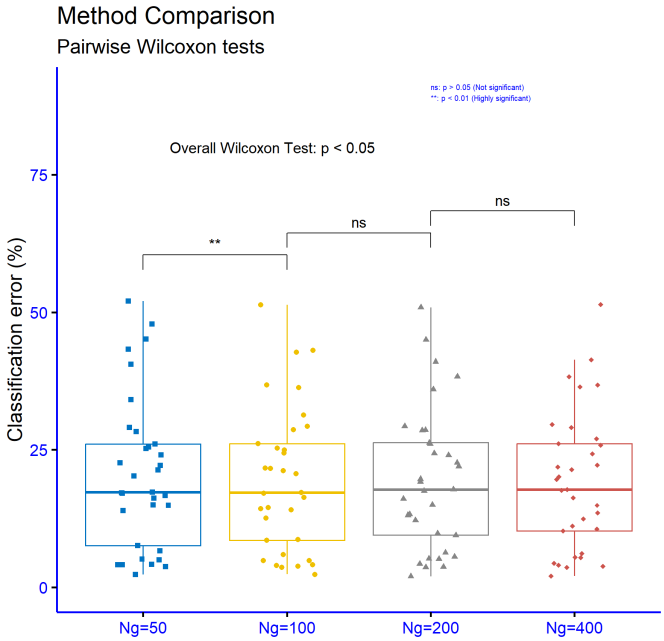

Figure 12. Statistical tests on the results obtained by the application of the current work on the classification datasets using a series of values for the parameter N_g.

In Figure 13, the p-value analysis on the regression datasets shows that the comparison between $N_g = 50$ and $N_g = 100$ is not statistically significant ($p > 0.05$, ns), so increasing generations in this range does not yield a consistent improvement. By contrast, moving from $N_g = 100$ to $N_g = 200$ is statistically significant ($p < 0.05$, *), indicating a measurable reduction in error around 200 generations. Finally, the comparison between $N_g = 200$ and $N_g = 400$ is not statistically significant ($p > 0.05$, ns), suggesting diminishing returns beyond 200 generations. Overall, the findings indicate that for regression problems the benefit concentrates around 200 generations, while further increases in N_g do not guarantee additional consistent gains.

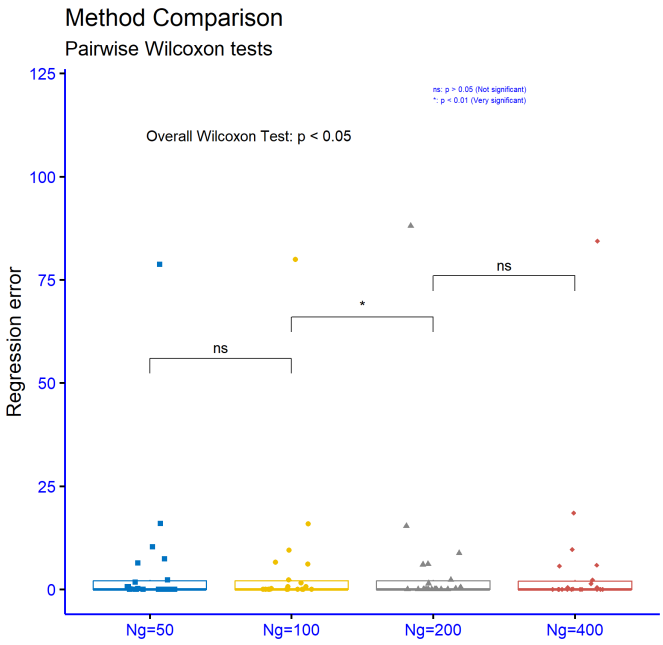

Figure 13. Statistical tests on the results obtained by the application of the current work on the regression datasets using a series of values for the parameter N_g

3.5. Experiments with real - world problems

One practical problem that can be considered here is the prediction of the duration of forest fires that was presented recently for the Greek territory[114]. Using data from the Greek Fire Service, an experiment was conducted for the prediction of the duration of forest fires for the years 2014-2024. In this experiment, the following methods were used:

1. A neural network with 10 computing nodes, trained using the BFGS optimizer.
2. A Radial Basis Function with 10 weights, trained with the original method for RBF training.
3. The proposed method.

The results for the prediction of the duration are graphically illustrated in Figure 14. As is evident from the graph, in almost all years the classification error of the proposed technique is lower than the other two machine learning techniques.

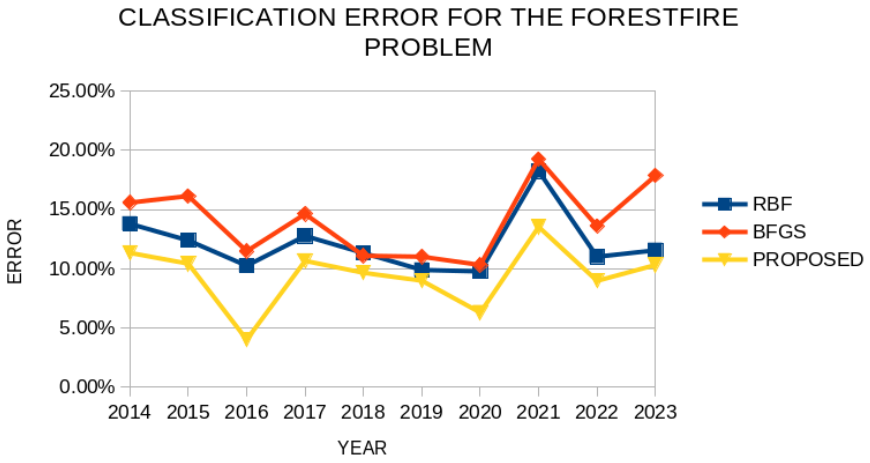

Figure 14. Average classification error for the years 2014-2023 for the forest fires in the Greek territory.

The second real - world example is the PIRvision dataset [115], which includes occupancy detection data and it has 15302 patterns. The dimension of each pattern is 59. The same machine learning models were also used in the case and the average classification error for these methods is depicted graphically in Figure 15. One more time, the proposed method has lower classification error than the other methods involved in this experiment.

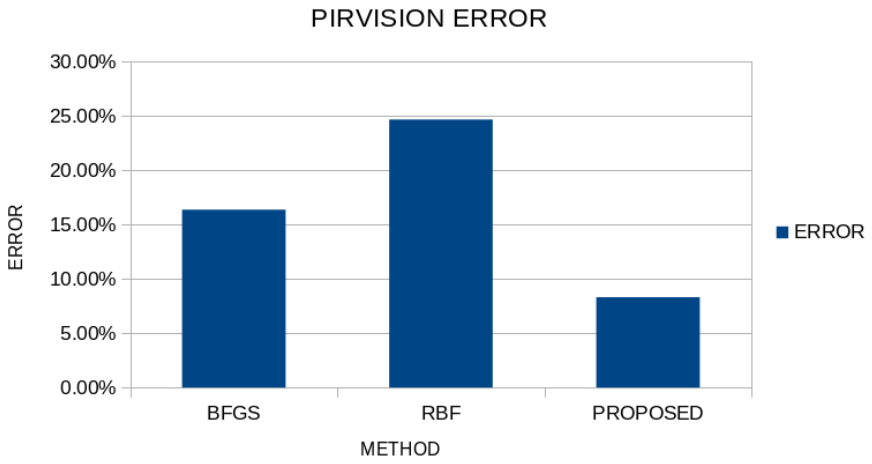

Figure 15. The experimental results for the PIRvision dataset.

4. Conclusions

The final experimental evidence shows that the three-phase RBF training pipeline bound construction via K-means, global search with a GA inside those bounds, and local refinement with BFGS yields robust gains across heterogeneous classification and regression tasks. On classification, it achieves the lowest mean error (19.45%) with extremely significant superiority over all baselines ($p < 0.0001$), on regression, it attains the smallest mean absolute error (5.87), with $p < 0.01$ against BFGS/ADAM and $p < 0.0001$ against NEAT/RBF-KMEANS/GENRBF. These results indicate that coupling broad exploration with constrained, precise local tuning mitigates numerical instability and local minima, providing reproducible performance improvements.

Sensitivity analyses reveal that the scale factor F materially affects classification at small-to-intermediate settings ($F = 1 \to 2$ and $F = 2 \to 4$ are significant at $p < 0.01$), with no meaningful gain from $F = 4$ to $F = 8$, whereas for regression the F comparisons are not significant, highlighting methodological stability. Alternative σ computation methods $\left(\sigma_1,\ \sigma_{\text{avg}},\ \sigma_{\text{max}}\right)$ differ only marginally on average and show no significant differences in either task, reinforcing the method's resilience to low-level design choices.

Automating architecture and hyperparameter adaptation is a natural next step. Joint optimization of the number of RBF units, F, and bounds via Bayesian optimization or meta-learning could reduce manual tuning and improve generalization. Exploring alternative global optimizers (e.g., DE, PSO, CMA-ES) or hybrid GA and Bayesian strategies may accelerate convergence and enhance exploration, while in the final stage L-BFGS, bound-aware variants, and stochastic formulations could benefit large-scale, high-dimensional settings. A thorough ablation study to quantify each phase's contribution, along with broader post-hoc statistics, would strengthen the evidence base. From a systems perspective, parallel/distributed GA evaluations and GPU-accelerated RBF computations can materially cut runtime. Finally, extending benchmarks to strong non-RBF baselines and integrating the approach into AutoML pipelines together with analyses of interpretability and predictive uncertainty will provide a more complete picture of the method's limits and potential.

However, it should be emphasized that, despite its efficiency, the proposed method demands more computational time than other machine learning techniques due to the serial application of the three training stages. In particular, the second stage of the genetic algorithm application requires significant computational time for its execution. However this time can be significantly reduced by using modern parallel computing techniques like OpenMP [116] or MPI [117].

Author Contributions: V.C. and I.G.T. conducted the experiments, employing several datasets and provided the comparative experiments. D.T. and 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.

Acknowledgments: This research has been financed by the European Union : Next Generation EU through the Program Greece 2.0 National Recovery and Resilience Plan , under the call RESEARCH – CREATE – INNOVATE, project name “iCREW: Intelligent small craft simulator for advanced crew training using Virtual Reality techniques" (project code:TAEDK-06195).

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

References

1. Raissi, M., & Karniadakis, G. E. (2018). Hidden physics models: Machine learning of nonlinear partial differential equations. Journal of Computational Physics, 357, 125-141.
2. Kashinath, K., Mustafa, M., Albert, A., Wu, J. L., Jiang, C., Esmaeilzadeh, S., ... & Prabhat, N. (2021). Physics-informed machine learning: case studies for weather and climate modelling. Philosophical Transactions of the Royal Society A, 379(2194), 20200093.
3. Viquar, M., Basak, S., Dasgupta, A., Agrawal, S., & Saha, S. (2019). Machine learning in astronomy: A case study in quasar-star classification. Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2018, Volume 3, 827-836.
4. Luo, S., Leung, A. P., Hui, C. Y., & Li, K. L. (2020). An investigation on the factors affecting machine learning classifications in gamma-ray astronomy. Monthly Notices of the Royal Astronomical Society, 492(4), 5377-5390.
5. Meuwly, M. (2021). Machine learning for chemical reactions. Chemical Reviews, 121(16), 10218-10239.
6. 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.
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. Athey, S. (2018). The impact of machine learning on economics. In The economics of artificial intelligence: An agenda (pp. 507-547). University of Chicago Press.
10. 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.
11. Ghai, D., Tripathi, S. L., Saxena, S., Chanda, M., & Alazab, M. (Eds.). (2022). Machine learning algorithms for signal and image processing. John Wiley & Sons.
12. Ahmed, N. K., Atiya, A. F., Gayar, N. E., & El-Shishiny, H. (2010). An empirical comparison of machine learning models for time series forecasting. Econometric reviews, 29(5-6), 594-621.
13. Radha, V., & Nallammal, N. (2011, October). Neural network based face recognition using RBFN classifier. In Proceedings of the world congress on engineering and computer science (Vol. 1, pp. 19-21).
14. Kumar, M., & Yadav, N. (2011). Multilayer perceptrons and radial basis function neural network methods for the solution of differential equations: a survey. Computers & Mathematics with Applications, 62(10), 3796-3811.
15. Zhang, Y. (2019). An accurate and stable RBF method for solving partial differential equations. Applied Mathematics Letters, 97, 93-98.
16. Dash, R., & Dash, P. K. (2015, October). A comparative study of radial basis function network with different basis functions for stock trend prediction. In 2015 IEEE Power, Communication and Information Technology Conference (PCITC) (pp. 430-435). IEEE.

17. R. -J. Lian, Adaptive Self-Organizing Fuzzy Sliding-Mode Radial Basis-Function Neural-Network Controller for Robotic Systems, IEEE Transactions on Industrial Electronics **61**, pp. 1493-1503, 2014.
18. M. Vijay, D. Jena, Backstepping terminal sliding mode control of robot manipulator using radial basis functional neural networks. Computers & Electrical Engineering **67**, pp. 690-707, 2018.
19. 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.
20. 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.
21. J. A. Leonard and M. A. Kramer, "Radial basis function networks for classifying process faults," in IEEE Control Systems Magazine, vol. 11, no. 3, pp. 31-38, April 1991, doi: 10.1109/37.75576.
22. Gan, M., Peng, H., & Dong, X. P. (2012). A hybrid algorithm to optimize RBF network architecture and parameters for nonlinear time series prediction. Applied Mathematical Modelling, 36(7), 2911-2919.
23. G. Sideratos and N. Hatziargyriou, "Using Radial Basis Neural Networks to Estimate Wind Power Production," 2007 IEEE Power Engineering Society General Meeting, Tampa, FL, USA, 2007, pp. 1-7, doi: 10.1109/PES.2007.385812.
24. MacQueen, J.: 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.
25. J. Park and I. W. Sandberg, "Universal Approximation Using Radial-Basis-Function Networks," in Neural Computation, vol. 3, no. 2, pp. 246-257, June 1991, doi: 10.1162/neco.1991.3.2.246.
26. L.I. Kuncheva, Initializing of an RBF network by a genetic algorithm, Neurocomputing **14**, pp. 273-288, 1997.
27. 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.
28. D. Wang, X.J. Zeng, J.A. Keane, A clustering algorithm for radial basis function neural network initialization, Neurocomputing **77**, pp. 144-155, 2012.
29. N. Benoudjit, M. Verleysen, On the Kernel Widths in Radial-Basis Function Networks, Neural Processing Letters **18**, pp. 139–154, 2003.
30. Määttä, J., Bazaliy, V., Kimari, J., Djurabekova, F., Nordlund, K., & Roos, T. (2021). Gradient-based training and pruning of radial basis function networks with an application in materials physics. Neural Networks, 133, 123-131.
31. Gale, S., Vestheim, S., Gravdahl, J. T., Fjerdingen, S., & Schjølberg, I. (2013, July). RBF network pruning techniques for adaptive learning controllers. In 9th International Workshop on Robot Motion and Control (pp. 246-251). IEEE.
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. Du, J. X., Huang, D. S., Zhang, G. J., & Wang, Z. F. (2006). A novel full structure optimization algorithm for radial basis probabilistic neural networks. Neurocomputing, 70(1-3), 592-596.
34. Yu, H., Reiner, P. D., Xie, T., Bartczak, T., & Wilamowski, B. M. (2014). An incremental design of radial basis function networks. IEEE transactions on neural networks and learning systems, 25(10), 1793-1803.
35. Ding, S., Xu, L., Su, C., & Jin, F. (2012). An optimizing method of RBF neural network based on genetic algorithm. Neural Computing and Applications, 21(2), 333-336.
36. Jia, W., Zhao, D., & Ding, L. (2016). An optimized RBF neural network algorithm based on partial least squares and genetic algorithm for classification of small sample. Applied Soft Computing, 48, 373-384.
37. Rani R, H. J., & Victoire T, A. A. (2018). Training radial basis function networks for wind speed prediction using PSO enhanced differential search optimizer. PloS one, 13(5), e0196871.
38. Zhang, W., & Wei, D. (2018). Prediction for network traffic of radial basis function neural network model based on improved particle swarm optimization algorithm. Neural Computing and Applications, 29(4), 1143-1152.
39. Qasem, S. N., Shamsuddin, S. M., & Zain, A. M. (2012). Multi-objective hybrid evolutionary algorithms for radial basis function neural network design. Knowledge-Based Systems, 27, 475-497.
40. 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.
41. 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.
42. Rani R, H. J., & Victoire T, A. A. (2018). Training radial basis function networks for wind speed prediction using PSO enhanced differential search optimizer. PloS one, 13(5), e0196871.
43. Karamichailidou, D., Gerolymatos, G., Patrinos, P., Sarimveis, H., & Alexandridis, A. (2024). Radial basis function neural network training using variable projection and fuzzy means. Neural Computing and Applications, 36(33), 21137-21151.
44. K. Krishna, M. Narasimha Murty, Genetic K-means algorithm, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **29**, pp. 433-439, 1999.

45. K. P. Sinaga, M. -S. Yang, Unsupervised K-Means Clustering Algorithm, IEEE Access **8**, pp. 80716-80727, 2020.
46. M. Ay, L. Özbakır, S. Kulluk, B. Gülmez , G.Öztürk, S. Özer, FC-Kmeans: Fixed-centered K-means algorithm, Expert Systems with Applications **211**, 118656, 2023.
47. E.U. Oti, M.O. Olusola, F.C. Eze, S.U. Enogwe, Comprehensive review of K-Means clustering algorithms, Criterion **12**, pp. 22-23, 2021.
48. S.A. Grady, M.Y. Hussaini, M.M. Abdullah, Placement of wind turbines using genetic algorithms, Renewable Energy **30**, pp. 259-270, 2005.
49. Parvaze, S., Kumar, R., Khan, J. N., Al-Ansari, N., Parvaze, S., Vishwakarma, D. K., ... & Kuriqi, A. (2023). Optimization of water distribution systems using genetic algorithms: A review. Archives of Computational Methods in Engineering, 30(7), 4209-4244.
50. Gordini, N. (2014). A genetic algorithm approach for SMEs bankruptcy prediction: Empirical evidence from Italy. Expert systems with applications, 41(14), 6433-6445.
51. Ding, S., Su, C., & Yu, J. (2011). An optimizing BP neural network algorithm based on genetic algorithm. Artificial intelligence review, 36(2), 153-162.
52. Guo, L., Funie, A. I., Thomas, D. B., Fu, H., & Luk, W. (2016). Parallel genetic algorithms on multiple FPGAs. ACM SIGARCH Computer Architecture News, 43(4), 86-93.
53. Johar, F. M., Azmin, F. A., Suaidi, M. K., Shibghatullah, A. S., Ahmad, B. H., Salleh, S. N., ... & Shukor, M. M. (2013, November). A review of genetic algorithms and parallel genetic algorithms on graphics processing unit (GPU). In 2013 IEEE International Conference on Control System, Computing and Engineering (pp. 264-269). IEEE.
54. P. Kaelo, M.M. Ali, Integrated crossover rules in real coded genetic algorithms, European Journal of Operational Research **176**, pp. 60-76, 2007.
55. M.J.D Powell, A Tolerant Algorithm for Linearly Constrained Optimization Calculations, Mathematical Programming **45**, pp. 547-566, 1989.
56. Liu, D. C., & Nocedal, J. (1989). On the limited memory BFGS method for large scale optimization. Mathematical programming, 45(1), 503-528.
57. A. Mokhtari and A. Ribeiro, "RES: Regularized Stochastic BFGS Algorithm," in IEEE Transactions on Signal Processing, vol. 62, no. 23, pp. 6089-6104, Dec.1, 2014, doi: 10.1109/TSP.2014.2357775.
58. Dai, Y. H. (2002). Convergence properties of the BFGS algoritm. SIAM Journal on Optimization, 13(3), 693-701.
59. M. Kelly, R. Longjohn, K. Nottingham, The UCI Machine Learning Repository, https://archive.ics.uci.edu (accessed on 28 September 2025).
60. 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. https://sci2s.ugr.es/keel/datasets.php (accessed on 28 September 2025).
61. Kooperberg, C. (1997). Statlib: an archive for statistical software, datasets, and information. The American Statistician, 51(1), 98. https://lib.stat.cmu.edu/datasets/ (accessed on 28 September 2025).
62. 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.
63. Tzimourta, K.D.; Tsoulos, I.; Bilero, I.T.; Tzallas, A.T.; Tsipouras, M.G.; Giannakeas, N. Direct Assessment of Alcohol Consumption in Mental State Using Brain Computer Interfaces and Grammatical Evolution. Inventions 2018, 3, 51.
64. J.R. Quinlan, Simplifying Decision Trees. International Journal of Man-Machine Studies **27**, pp. 221-234, 1987.
65. T. Shultz, D. Mareschal, W. Schmidt, Modeling Cognitive Development on Balance Scale Phenomena, Machine Learning **16**, pp. 59-88, 1994.
66. 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.
67. R. Setiono , W.K. Leow, FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks, Applied Intelligence **12**, pp. 15-25, 2000.
68. Gavrilis, D., Tsoulos, I. G., & Dermatas, E. (2008). Selecting and constructing features using grammatical evolution. Pattern Recognition Letters, 29(9), 1358-1365.
69. 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.
70. P. Horton, K.Nakai, A Probabilistic Classification System for Predicting the Cellular Localization Sites of Proteins, In: Proceedings of International Conference on Intelligent Systems for Molecular Biology **4**, pp. 109-15, 1996.
71. 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.
72. I. Kononenko, E. Šimec, M. Robnik-Šikonja, Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF, Applied Intelligence **7**, pp. 39–55, 1997

73. Rashid, Tarik A.; Hassan, Bryar (2022), "Heart Attack Dataset", Mendeley Data, V1, doi: 10.17632/wmhctcrt5v.1
74. 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.
75. J.G. Dy , C.E. Brodley, Feature Selection for Unsupervised Learning, The Journal of Machine Learning Research **5**, pp 845–889, 2004.
76. S. J. Perantonis, V. Virvilis, Input Feature Extraction for Multilayered Perceptrons Using Supervised Principal Component Analysis, Neural Processing Letters **10**, pp 243–252, 1999.
77. J. Garcke, M. Griebel, Classification with sparse grids using simplicial basis functions, Intell. Data Anal. **6**, pp. 483-502, 2002.
78. J. Mcdermott, R.S. Forsyth, Diagnosing a disorder in a classification benchmark, Pattern Recognition Letters **73**, pp. 41-43, 2016.
79. 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.
80. 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.
81. M.A. Little, P.E. McSharry, S.J Roberts et al, Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection. BioMed Eng OnLine **6**, 23, 2007.
82. M.A. Little, P.E. McSharry, E.J. Hunter, J. Spielman, L.O. Ramig, Suitability of dysphonia measurements for telemonitoring of Parkinson's disease. IEEE Trans Biomed Eng. **56**, pp. 1015-1022, 2009.
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. N. Giannakeas, M.G. Tsipouras, A.T. Tzallas, K. Kyriakidi, Z.E. Tsianou, P. Manousou, A. Hall, E.C. Karvounis, V. Tsianos, E. Tsianos, 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. Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.
89. P. Cortez, A. M. Gonçalves Silva, Using data mining to predict secondary school student performance, In Proceedings of 5th FUture BUsiness TEChnology Conference (FUBUTEC 2008) (pp. 5–12). EUROSIS-ETI, 2008.
90. I-Cheng Yeh, King-Jang Yang, Tao-Ming Ting, Knowledge discovery on RFM model using Bernoulli sequence, Expert Systems with Applications **36**, pp. 5866-5871, 2009.
91. Jeyasingh, S., & Veluchamy, M. (2017). Modified bat algorithm for feature selection with the Wisconsin diagnosis breast cancer (WDBC) dataset. Asian Pacific journal of cancer prevention: APJCP, 18(5), 1257.
92. Alshayeji, M. H., Ellethy, H., & Gupta, R. (2022). Computer-aided detection of breast cancer on the Wisconsin dataset: An artificial neural networks approach. Biomedical signal processing and control, 71, 103141.
93. 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.
94. P. Zhong, M. Fukushima, Regularized nonsmooth Newton method for multi-class support vector machines, Optimization Methods and Software **22**, pp. 225-236, 2007.
95. 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," Physical Review E, vol. 64, no. 6, Article ID 061907, 8 pages, 2001.
96. A. T. Tzallas, M. G. Tsipouras, and D. I. Fotiadis, "Automatic Seizure Detection Based on Time-Frequency Analysis and Artificial Neural Networks," Computational Intelligence and Neuroscience, vol. 2007, Article ID 80510, 13 pages, 2007. doi:10.1155/2007/80510
97. M. Koivisto, K. Sood, Exact Bayesian Structure Discovery in Bayesian Networks, The Journal of Machine Learning Research **5**, pp. 549–573, 2004.

98. 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; ISSN 1034-3288
99. Brooks, T.F.; Pope, D.S.; Marcolini, A.M. Airfoil Self-Noise and Prediction. Technical Report, NASA RP-1218. July 1989. Available online: https://ntrs.nasa.gov/citations/19890016302 (accessed on 14 November 2024).
100. Quinlan,R. (1993). Combining Instance-Based and Model-Based Learning. In Proceedings on the Tenth International Conference of Machine Learning, 236-243, University of Massachusetts, Amherst. Morgan Kaufmann.
101. I.Cheng Yeh, Modeling of strength of high performance concrete using artificial neural networks, Cement and Concrete Research. **28**, pp. 1797-1808, 1998.
102. Friedman, J. (1991): Multivariate Adaptative Regression Splines. Annals of Statistics, 19:1, 1--141.
103. D. Harrison and D.L. Rubinfeld, Hedonic prices and the demand for clean ai, J. Environ. Economics & Management **5**, pp. 81-102, 1978.
104. I.G. Tsoulos, V. Charilogis, G. Kyrou, V.N. Stavrou, A. Tzallas, Journal of Open Source Software **10**, 7584, 2025.
105. Yuan, Y. X. (1991). A modified BFGS algorithm for unconstrained optimization. IMA Journal of Numerical Analysis, 11(3), 325-332.
106. C. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.
107. G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of Control Signals and Systems **2**, pp. 303-314, 1989.
108. D. P. Kingma, J. L. Ba, ADAM: a method for stochastic optimization, in: Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015), pp. 1–15, 2015.
109. Y. Xue, Y. Tong, F. Neri, An ensemble of differential evolution and Adam for training feed-forward neural networks. Information Sciences **608**, pp. 453-471, 2022.
110. K. O. Stanley, R. Miikkulainen, Evolving Neural Networks through Augmenting Topologies, Evolutionary Computation **10**, pp. 99-127, 2002.
111. Ward, R., Wu, X., & Bottou, L. (2020). Adagrad stepsizes: Sharp convergence over nonconvex landscapes. Journal of Machine Learning Research, 21(219), 1-30.
112. Ruben Martinez-Cantin, BayesOpt: A Bayesian Optimization Library for Nonlinear Optimization, Experimental Design and Bandits. Journal of Machine Learning Research, 15(Nov):3735--3739, 2014.
113. S. Ding, L. Xu, C. Su et al, An optimizing method of RBF neural network based on genetic algorithm. Neural Comput & Applic 21, pp. 333–336, 2012.
114. C. Kopitsa, I.G. Tsoulos, V. Charilogis, A. Stavrakoudis, Predicting the Duration of Forest Fires Using Machine Learning Methods, Future Internet **16**, 396, 2024.
115. Emad-Ud-Din, M., & Wang, Y. (2023). Promoting occupancy detection accuracy using on-device lifelong learning. IEEE Sensors Journal, 23(9), 9595-9606.
116. Saxena, R., Jain, M., Malhotra, K., & Vasa, K. D. (2019). An optimized openmp-based genetic algorithm solution to vehicle routing problem. In Smart Computing Paradigms: New Progresses and Challenges: Proceedings of ICACNI 2018, Volume 2 (pp. 237-245). Singapore: Springer Singapore.
117. Rajan, S. D., & Nguyen, D. T. (2004). Design optimization of discrete structural systems using MPI-enabled genetic algorithm. Structural and Multidisciplinary Optimization, 28(5), 340-348.