MDPI

Article

Introducing a new genetic operator based on Differential Evolution for effective training of neural networks

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; tsalikakis@gmail.com
* Correspondence: itsoulos@uoi.gr

Abstract: Artificial neural networks are widely established models used in a variety of real - world problems derived from physics, chemistry etc. These machine learning models contain a series of parameters that must be appropriately tuned by various optimization techniques in order to be effective in the problems they face. Genetic algorithms have been used in many cases in the recent literature to train artificial neural networks and various modifications have been introduced to enhance this procedure. In this article, the incorporation of a novel genetic operator in genetic algorithms is proposed in order to effectively train artificial neural networks. The new operator is based on the differential evolution technique and it is periodically applied to randomly selected chromosomes from the genetic population. Furthermore, to find a promising range of values for the parameters of the artificial neural network, an additional genetic algorithm is executed before the execution of the basic algorithm. The modified genetic algorithm was used to train neural networks for classification and regression datasets and the results are reported and compared against other methods that train neural networks.

Keywords: Neural networks; Genetic algorithms; Evolutionary computation

Citation: Tsoulos, I.G.; Charilogis, V.; Tsalikakis D. Introducing a new genetic operator based on Differential Evolution for effective training of neural networks. *Journal Not Specified* **2023**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

1. Introduction

A machine learning model that has been widely used in recent decades in dozens of problems are artificial neural networks [1,2], which are parametric models commonly defined as $N(\overrightarrow{x}, \overrightarrow{w})$. The vector $\overrightarrow{x}$ stands for the input pattern and the vector $\overrightarrow{w}$ represents the associated set of parameters that should be calculated by any optimization method. The calculation is performed by minimizing the so - called training error, expressed as:

$$E(N(\overrightarrow{x}, \overrightarrow{w})) = \sum_{i=1}^{M} (N(\overrightarrow{x}_i, \overrightarrow{w}) - y_i)^2 \quad (1)$$

The values $(\overrightarrow{x_i}, y_i)$, $i = 1, ..., M$ form the training set of the problem, where y_i represent the expected outputs for each pattern $\overrightarrow{x_i}$.

Artificial neural networks have applied on a wide series of problems from various areas, such as physics [3,4], astronomy [5], chemistry [6], economics [7], medicine [8,9] etc. The equation 1 has been minimized by various methods in the relevant literature. Among them one can find the Back propagation method [10,11], the RPROP method [12–14], Quasi Newton methods [15,16], Simulated Annealing [17], Particle Swarm Optimization (PSO) [18, 19], Genetic Algorithms [20,21], Differential Evolution [22], Ant Colony Optimization [23], Gray Wolf Optimizer [24], Whale optimization [25] etc. Moreover, Zhang et al proposed a hybrid algorithm that conjuncts PSO and the Back Propagation algorithm for neural networks training [26]. Also, recently many researchers proposed methods that take advantage of parallel processing units in order to speed up the training process [27,28].

Moreover, recently Kang et al proposed a hybrid method that combines lattice Boltzmann method [29] and various machine learning methods with the neural networks included in them with good approximation abilities [30]. Furthermore, a series of papers has been published recently that tackle the initialization procedure for the parameters of neural networks. These methods include decision trees [31], incorporation of the Cauchy's inequality [32], discriminant learning [33], usage of polynomial bases [34], usage of intervals [35] etc. A systematic review of initialization methods can be found in the work of Narkhede et al [36].

Additionally, finding the optimal architecture of an artificial neural network can effectively contribute to its training, since on the one hand it will reduce the required training time and on the other hand it will eliminate the problem of overfitting. In this direction, a series of researchers have proposed many methods to tackle this problem, such as genetic algorithms [37,38], the application of the PSO method method [39], application of reinforcement learning [40] etc. Also, Tsoulos et al. proposed the usage of Grammatical Evolution technique [41] to construct artificial neural networks [42].

This paper proposes the usage of a two-stage technique for the efficient training of artificial neural networks. In the first stage, a genetic algorithm is used to efficiently identify a range of values within which the parameters of the artificial neural network should be optimized. In the second stage, a genetic algorithm optimizes these parameters, which uses a new operator to enhance its results. This new operator is based on the differential evolution technique [43] and is applied periodically to randomly selected chromosomes of the genetic population. The first stage of the technique is necessary to ensure that the parameters of the artificial neural network will be trained within a range of values, which will prevent their overfitting as much as possible. In the second stage, the differential evolution method was selected as the base of the new operator. This method is an evolutionary technique widely used in a series of practical problems, such as community detection [44], structure prediction [45], motor fault diagnosis [46], and clustering techniques [47]. Furthermore, this method was chosen as the basis for the new genetic operator due to the small number of required parameters that the user must specify.

The remaining of this article is organized as follows: in section 2 the proposed method is discussed in detail, in section 3 the used datasets as well as the conducted experiments are discussed and finally, in section 4 some conclusions are presented.

2. Method description

The two phases of the proposed method are analyzed in detail in this section. During the first phase a genetic algorithm is utilized in order to detect a promising interval of values for the parameters of neural network. In the second phase a genetic algorithm that incorporates the suggested operator is applied to minimize the training error of the neural network and the parameters are initialized inside the interval located during the first phase.

2.1. The first phase of the proposed method

In the first phase of the proposed technique, a genetic algorithm is used to identify a range of values for the parameters of the artificial neural network. Genetic algorithms are evolutionary methods, where a series of randomly created candidate solutions, those called chromosomes, are evolved ireratively through a series of steps similar to natural processes such as selection, crossover and mutation. Genetic algorithms have been used successfully in a series of real - world problems, such as placement of wind turbines [48], water distribution [49], economics [50], neural network training [51] etc. The neural networks adopted in this manuscript have the following form, as proposed in [42]:

$$N(\overrightarrow{x}, \overrightarrow{w}) = \sum_{i=1}^{H} w_{(d+2)i-(d+1)} \sigma \left(\sum_{j=1}^{d} x_j w_{(d+2)i-(d+1)+j} + w_{(d+2)i} \right) \quad (2)$$

Where the value H denotes the total number of processing units in this network and the value d defines the number of inputs for the pattern $\overrightarrow{x}$. Hence, the total number of parameters for this network are: $n = (d+2)H$. The function $\sigma(x)$ is the sigmoid function, defined as:

$$\sigma(x) = \frac{1}{1+\exp(-x)} \tag{3}$$

The steps of the algorithm of the first phase have as follows:

1. **Initialization step**.
   (a) **Set** the number of chromosomes N_c and the maximum number of allowed generations N_g.
   (b) **Set** the selection rate p_s and the mutation rate p_m.
   (c) **Set** the margin factor a, where $a \geq 1$.
   (d) **Set** $k = 0$ as the generation counter.
   (e) **Initialize** randomly the chromosomes $g_i,\ i = 1, \ldots, N_c$. Each chromosome is a vector of parameters for the artificial neural network.
2. **Fitness calculation step**.
   (a) **For** $i = 1, \ldots, N_c$ **do**
      i. **Create** the neural network $N_i(\overrightarrow{x}, \overrightarrow{g_i})$ for the chromosome g_i.
      ii. **Calculate** the associated fitness value f_i as

$$f_i = \sum_{j=1}^{M} \left(N_i(\overrightarrow{x_j}, \overrightarrow{g_i}) - y_j\right)^2$$

      for the pairs $(\overrightarrow{x_j}, y_j),\ j = 1, \ldots, M$ of the training set.
   (b) **End For**
3. **Genetic operations step.**
   (a) **Transfer** the best $(1 - p_s) \times N_c$ chromosomes of the current generation to the next one. The remaining will be replaced by chromosomes produced in crossover and mutation.
   (b) **Perform** the crossover procedure. During this procedure, for each pair of constructed chromosomes $(\widetilde{z}, \widetilde{w})$ two chromosomes will be selected from the current population using tournament selection. The production of the new chromosomes is performed using process suggested by Kaelo et al [52].
   (c) **Perform** the mutation procedure. During the mutation procedure, for each element of each chromosome a random number $r \in [0,1]$ is selected. The corresponding element is altered randomly when $r \leq p_m$.
4. **Termination check step**.
   (a) **Set** $k = k + 1$
   (b) **If** $k \leq N_g$ then goto Fitness Calculation step.
5. **Margin creation step**.
   (a) **Obtain** the best chromosome g^* with the lowest fitness value.
   (b) **Create** the vectors $L^*$and R^* as:

$$\begin{aligned} L_i^* &= -a\left|g_i^*\right|,\ i = 1, \ldots, n \\ R_i^* &= a\left|g_i^*\right|,\ i = 1, \ldots, n \end{aligned}$$

2.2. The second phase of the proposed method

During the second phase a second genetic algorithm is used to minimize the training error of the neural network. The parameters of the neural network are initialized inside the vectors $L^*$and R^* produced in the previous phase of the algorithm. Also, a novel

stochastic genetic operator, which is based on the Differential Evolution approach, is applied periodically to the genetic population. This new stochastic operator is used to improve the performance of randomly selected chromosomes and to speed up the overall genetic algorithm in finding the global minimum. The main steps of the algorithm executed on the second phase have as follows:

1. **Initialization step**.
   (a) **Set** the number of chromosomes N_c and the maximum number of allowed generations N_g.
   (b) **Set** the selection rate $p_s \leq 1$ and the mutation rate $p_m \leq 1$.
   (c) **Set** the crossover probability CR, used in the new genetic operator.
   (d) **Set** the differential weight F that will be used in the novel genetic operator.
   (e) **Set** as N_i the number of generations before the application of the new operator.
   (f) **Set** as N_l the number of chromosomes that will participate in the new operator.
   (g) **Initialize** the g_i, $i = 1, \ldots, N_c$ chromosomes inside the vectors $L^*$and R^* of the previous phase.
   (h) **Set** $k = 0$ the generation counter.
2. **Fitness calculation step**.
   (a) **For** $i = 1, \ldots, N_c$ **do**
      i. **Produce** the corresponding neural network $N_i(\overrightarrow{x}, \overrightarrow{g_i})$ for the chromosome g_i.
      ii. **Calculate** the fitness value f_i as

$$f_i = \sum_{j=1}^{M} \left(N_i(\overrightarrow{x_j}, \overrightarrow{g_i}) - y_j\right)^2$$

   (b) **End For**
3. **Application of genetic operators.**
   (a) **Copy** the best $(1 - p_s) \times N_c$ chromosomes with the lowest fitness values to the next generation. The remaining will be replaced by chromosomes produced in crossover and mutation.
   (b) **Apply** the same crossover procedure as in the algorithm of the first phase.
   (c) **Apply** the same mutation procedure as in the genetic algorithm of the first phase.
4. **Application of the novel genetic operator.**
   (a) **If** $k \bmod N_i = 0$ **then**
      i. **Create** the set $C = \{z_1, z_2, \ldots, z_{N_{cr}}\}$ of N_l randomly selected chromosomes.
      ii. **For** $i = 1, \ldots, N_l$ apply the deOperator of algorithm 1 to every chromosome $z_i \in C$.
   (b) **End if**
5. **Termination check step**.
   (a) **Set** $k = k + 1$
   (b) **If** $k \leq N_g$ goto Fitness calculation step.
6. **Testing step**.
   (a) **Obtain** the best chromosome g^* from the genetic population.
   (b) **Create** the corresponding neural network $N^*\left(\overrightarrow{x_j}, \overrightarrow{g^*}\right)$.
   (c) **Apply** this neural network to the test set of the objective problem and report the error.

Algorithm 1 The proposed genetic operator.

Function deOperator(g, F, CR)

1. **Select** three distinct chromosomes a, b, c from the current population using tournament selection.
2. **Set** $R \in [1, n]$ a randomly selected integer.
3. **Set** $t = g$, as the trial chromosome.
4. **For** $i = 1, \ldots, n$ **do**
   (a) **Select** $r \in [0, 1]$ a random number.
   (b) **If** $i =$R or $r \leq$ CR **then** $t_i = a_i + F \times (b_i - c_i)$
   (c) **Set** $t_f = \sum_{j=1}^{M} \left(N_i\left(\overrightarrow{x_j}, \overrightarrow{t}\right) - y_j \right)^2$
   (d) **If** $t_f \leq f_g$ then $g = t$.
5. **End For**
6. **Return** g.

End Function

The steps of all phases are also graphically illustrated in Figure 1.

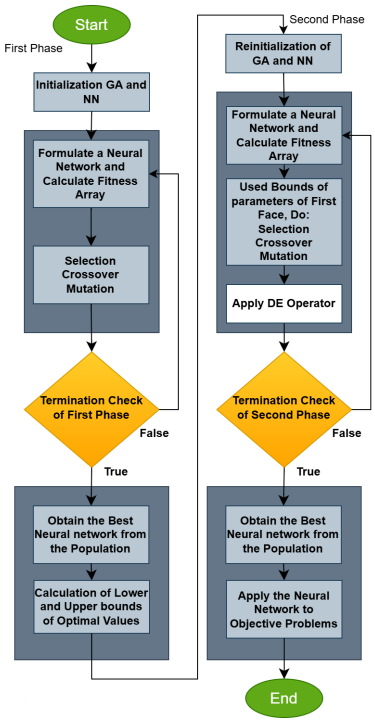

Figure 1. The flowchart of the proposed method.

3. Experiments

To demonstrate the dynamics and reliability of the proposed methodology, a series of experiments were carried out on known datasets from the relevant literature. These datasets were obtained from the following databases:

1. The UCI database https://archive.ics.uci.edu/(accessed on 5 March 2025)[53]
2. The Keel website, https://sci2s.ugr.es/keel/datasets.php(accessed on 5 March 2025)[54].
3. The Statlib URL ftp://lib.stat.cmu.edu/datasets/index.html(accessed on 5 March 2025).

3.1. Experimental datasets

The following series of classification datasets were used in the conducted experiments:

1. The Alcohol dataset, which is related to experiments on alcohol consumption [55].

2. The Appendicitis dataset, which is a medical dataset [56].
3. The Australian dataset, which is used in bank transactions [57].
4. The Balance dataset, which contains measurements from various psychological experiments [58].
5. The Circular dataset, which was created artificially.
6. The Cleveland dataset, which is a medical dataset [59,60].
7. The Dermatology dataset, which is a medical dataset regarding dermatology problems [61].
8. The Ecoli dataset, which is used in protein problems [62].
9. The Fert dataset, related to the detection of relations between sperm concentration and demographic data.
10. The Haberman dataset, which is related to the detection of breast cancer.
11. The Hayes roth dataset [63].
12. The Heart dataset, which is related to some heart diseases [64].
13. The HouseVotes dataset, related to data from Congressional voting in USA [65].
14. The Ionosphere dataset, that contains measurements from the ionosphere [66,67].
15. The Liverdisorder dataset, which is a medical dataset [68,69].
16. The Lymography dataset [70].
17. The Mammographic dataset, which is a medical dataset [71].
18. The Parkinsons dataset, that was used in the detection of Parkinson's disease [72,73].
19. The Pima dataset, a medical dataset related to the detection of diabetes's disease [74].
20. The Popfailures dataset, related to climate model simulations [75].
21. The Regions2 dataset, related to some diseases in liver [76].
22. The Saheart dataset, related to some heart diseases [77].
23. The Segment dataset, related to image processing [78].
24. The Sonar dataset, used to discriminate sonar signals [79].
25. The Spiral dataset, which was created artificially.
26. The StatHeart dataset, a medical dataset regarding heart diseases.
27. The Student dataset, which is related to experiments conducted in schools [80].
28. The WDBC dataset, which is related to the detection of cancer [81].
29. The Wine dataset, used to detection of the quality of wines [82,83].
30. The EEG dataset, which contains various EEG measurements [84,85]. From this dataset the following cases were utilized: Z_F_S, ZO_NF_S and ZONF_S.
31. The ZOO dataset, which is used for animal classification [86].

Also, the following regression datasets were incorporated in the conducted experiments:

1. The Abalone dataset, that was used to predict the age of abalones [87].
2. The Airfoil dataset, derived from NASA [88].
3. the Baseball dataset, used to predict the salary of baseball players.
4. The BK dataset, related to basketball games [89].
5. The BL dataset, related to some electricity experiments.
6. The Concrete dataset, which is related to civil engineering [90].
7. The Dee dataset, which is related to the price of electricity.
8. The Housing dataset, related to the price of houses [91].
9. The Friedman dataset, used in various benchmarks [92].
10. The FY dataset, related to fruit flies.
11. The HO dataset, obtained from the STATLIB repository.
12. The Laser dataset, related to laser experiments.
13. The LW dataset, related to the prediction of the weight of babes.
14. The MB dataset, which was obtained from Smoothing Methods in Statistics.
15. The Mortgage dataset, which is an economic dataset.
16. The Plastic dataset, related to the pressure in plastics.
17. The PY dataset [93].
18. The PL dataset, obtained from the STATLIB repository.
19. The Quake dataset, used to detect the strength of earthquakes.

20. The SN dataset, which is related to trellising and pruning.
21. The Stock dataset, used to estimate the price of stocks.
22. The Treasury dataset, which is an economic dataset.
23. The VE dataset, obtained from the STATLIB repository.

3.2. Experimental results

The code used in the conducted experiments was written in ANSI C++ and all runs were performed 30 times using a different seed for the random number generator each time. The validation of the experiments were done using the well - known method of 10 - fold cross validation. All the experiments were conducted on an AMD RYZEN 5950X machine with 128GB of ram running Debian Linux. For the case of classification datasets the average classification error is reported in the experimental tables. This error is calculated through the following equation:

$$E_C(N(w,x)) = 100 \times \frac{\sum_{i=1}^{K}(\text{class}(N(w,x_i)) - y_i)}{K} \tag{4}$$

where the set $T = \{x_i, y_i\}$, $i = 1, \ldots, K$ denotes the test set of the objective problem. For regression datasets the average regression error as calculated on the test set is reported and it is denoted as:Also, the regression error is defined as:

$$E_R(N(w,x)) = \frac{\sum_{i=1}^{K}(N(w,x_i) - y_i)^2}{K} \tag{5}$$

The values for the parameters of the proposed method are shown in Table 1. In the experimental tables the following notation is used:

1. The column DATASET represents the objective problem.
2. The column ADAM denotes the incorporation of the ADAM optimizer [94] to train a neural network with $H = 10$ processing nodes.
3. The column BFGS represents the application of the BFGS optimizer [95] to train a neural network with $H = 10$ processing nodes.
4. The column GENETIC denotes the usage of a Genetic Algorithm with the same set of parameters as shown in Table 1 to train an artificial neural network with $H = 10$ processing nodes.
5. The column NEAT is used for the application of the NEAT method (NeuroEvolution of Augmenting Topologies) [96].
6. The row AVERAGE is used for the average classification or regression error for all datasets.

Also, for the comparisons between methods, the Wilcoxon signed-rank test was used, a non-parametric test for paired data. Each method ("PROPOSED", "BFGS", etc.) was evaluated on the same dataset (dependent measurements), which is why a paired observations test was selected. The Mann-Whitney U test or t-test were not applied, as the former is for independent groups and the latter requires normally distributed data conditions not met here due to the use of a non-parametric test.

Table 1. The values of the experimental parameters.

PARAMETER	MEANING	VALUE
N_g	Number of generations allowed	200
N_c	Number of chromosomes	500
N_i	Number of generations before the application of the operator	20
N_l	Number of chromosomes where the operator will be applied	20
F	Differential Weight	0.8
CR	Crossover probability	0.9
p_s	Selection rate	0.1
p_m	Mutation rate	0.05
H	Number of processing nodes	10
a	Margin factor	1.0

The experimental results for the classification datasets are shown in Table 2 and for the regression datasets in Table 3.

Table 2. Experimental results for the classification datasets mentioned here using a series of machine learning methods. The numbers in cells represent average classification error as reported in the corresponding test set.

DATASET	ADAM	BFGS	NEAT	GENETIC	PROPOSED
Alcohol	57.78%	41.50%	66.80%	39.57%	24.79%
Appendicitis	16.50%	18.00%	17.20%	18.10%	15.97%
Australian	35.65%	38.13%	31.98%	32.21%	31.76%
Balance	7.87%	8.64%	23.14%	8.97%	8.39%
Circular	19.95%	6.08%	35.18%	5.99%	3.69%
Cleveland	67.55%	77.55%	53.44%	51.60%	48.10%
Dermatology	26.14%	52.92%	32.43%	30.58%	7.74%
Ecoli	64.43%	69.52%	43.44%	54.67%	47.62%
Fert	23.98%	23.20%	15.37%	28.50%	22.00%
Haberman	29.00%	29.34%	24.04%	28.66%	25.99%
Hayes Roth	59.70%	37.33%	50.15%	56.18%	37.00%
Heart	38.53%	39.44%	39.27%	28.34%	24.79%
HouseVotes	7.48%	7.13%	10.89%	6.62%	5.22%
Ionosphere	16.64%	15.29%	19.67%	15.14%	9.56%
Liverdisorder	41.53%	42.59%	30.67%	31.11%	31.08%
Lymography	29.26%	35.43%	33.70%	23.26%	28.60%
Mammographic	46.25%	17.24%	22.85%	19.88%	16.98%
Parkinsons	24.06%	27.58%	18.56%	18.05%	18.02%
Pima	34.85%	35.59%	34.51%	32.19%	30.44%
Popfailures	5.18%	5.24%	7.05%	5.94%	4.29%
Regions2	29.85%	36.28%	33.23%	29.39%	26.43%
Saheart	34.04%	37.48%	34.51%	34.86%	32.60%
Segment	49.75%	68.97%	66.72%	57.72%	30.00%
Sonar	30.33%	25.85%	34.10%	22.40%	18.78%
Spiral	47.67%	47.99%	48.66%	48.66%	44.20%
Statheart	44.04%	39.65%	44.36%	27.25%	22.72%
Student	5.13%	7.14%	10.20%	5.61%	4.16%
Wdbc	35.35%	29.91%	12.88%	8.56%	7.73%
Wine	29.40%	59.71%	25.43%	19.20%	8.55%
Z_F_S	47.81%	39.37%	38.41%	10.73%	6.46%
ZO_NF_S	47.43%	43.04%	43.75%	21.54%	6.01%
ZONF_S	11.99%	15.62%	5.44%	2.60%	1.79%
ZOO	14.13%	10.70%	20.27%	16.67%	9.07%
AVERAGE	**32.70%**	**33.01%**	**31.16%**	**25.48%**	**20.02%**

Table 3. Experimental results for the regression datasets using a series of machine learning methods. Numbers in cells stand for the average regression error as calculated in the corresponding test set.

DATASET	ADAM	BFGS	GENETIC	NEAT	PROPOSED
ABALONE	4.30	5.69	7.17	9.88	4.33
AIRFOIL	0.005	0.003	0.003	0.067	0.003
BASEBALL	77.90	119.63	103.60	100.39	67.45
BK	0.03	0.28	0.027	0.15	0.02
BL	0.28	2.55	5.74	0.05	0.002
CONCRETE	0.078	0.066	0.0099	0.081	0.003
DEE	0.63	2.36	1.013	1.512	0.20
HOUSING	80.20	97.38	43.26	56.49	26.62
FRIEDMAN	22.90	1.263	1.249	19.35	1.33
FY	0.038	0.22	0.65	0.08	0.039
HO	0.035	0.62	2.78	0.169	0.014
LASER	0.03	0.015	0.59	0.084	0.0027
LW	0.028	2.98	1.90	0.17	0.016
MB	0.06	0.129	3.39	0.061	0.048
MORTGAGE	9.24	8.23	2.41	14.11	0.31
PLASTIC	11.71	20.32	2.79	20.77	2.20
PL	0.117	0.29	0.28	0.098	0.023
PY	0.09	0.578	105.41	0.075	0.016
QUAKE	0.06	0.42	0.040	0.298	0.043
SN	0.206	0.40	2.95	0.174	0.024
STOCK	180.89	302.43	3.88	215.82	3.47
TREASURY	11.16	9.91	2.929	15.52	0.44
VE	0.359	1.92	2.43	0.045	0.023
AVERAGE	**17.41**	**25.12**	**12.80**	**19.80**	**4.64**

Also, in Figures 2 and 3 the statistical comparison for the experimental results is outlined graphically.

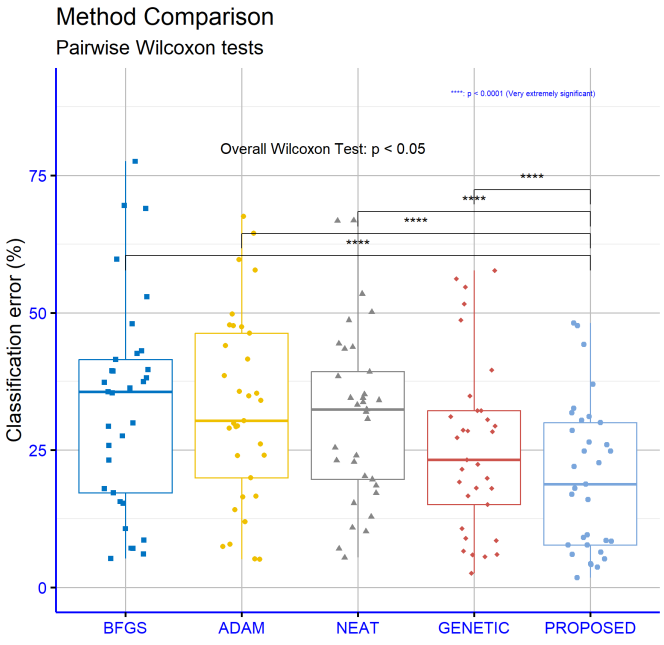

Figure 2. Statistical comparison for the obtained experimental results using a variety of machine learning methods in the classification datasets.

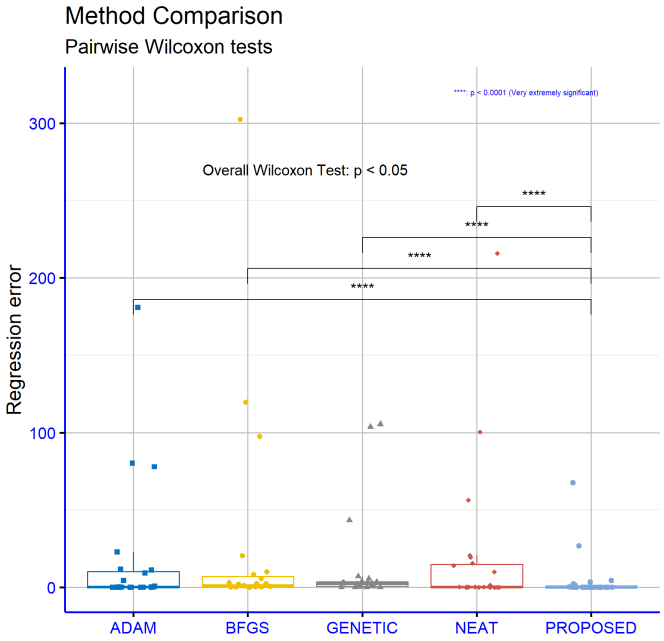

Figure 3. Statistical comparison for the experimental results using a series of machine learning methods on the regression datasets.

In the scientific analysis of the experimental results, the proposed model (PROPOSED) demonstrates statistically significant superiority over other methods (ADAM, BFGS, GENETIC, NEAT, RBF) in both classification and regression datasets. For classification, the PROPOSED model achieves a mean error rate of 20.02%, compared to 25.48%-33.01% for the other methods, with extremely low p-values (e.g., $p = 2.3 \times 10^{-10}$ against BFGS). In regression, the model's mean absolute error (4.64) is significantly lower than that of the comparative methods (12.8-25.12), with strong statistical significance $\left(p < 10^{-5}\right)$. However, in certain datasets (e.g., CLEVELAND, FRIEDMAN), the model's performance decreases, indicating dependence on data characteristics.

In the analysis of the experiments (Figure 2) for classification datasets, the proposed model (PROPOSED) demonstrates statistically significant superiority over all comparative methods (BFGS, ADAM, NEAT, GENETIC), with extremely low p-values ($p = 2.3 \times 10^{-10}$ to $p = 4.4 \times 10^{-6}$) . This indicates strong differences at a confidence level >99.9%, particularly against BFGS and ADAM $\left(p < 10^{-9}\right)$. For regression datasets (Figure 3), PROPOSED maintains significant superiority over all methods, though with slightly higher p-values $\left(p = 10^{-5}\right.$ to $\left.p = 2.4 \times 10^{-7}\right)$. The largest difference is observed against NEAT $\left(p = 2.4 \times 10^{-7}\right)$, while the smallest is against GENETIC $\left(p = 8.6 \times 10^{-5}\right)$.

Moreover, in Figure 4the a comparison of execution times is graphically outlined between the simple Genetic algorithm and the proposed method for various values of the critical parameter N_I. The dataset used in this experiment is the WINE classification dataset. As expected, increasing the value for this parameter also leads to a reduction in the required execution time for the proposed method, since the new genetic operator is applied more and more sparsely to the population chromosomes.

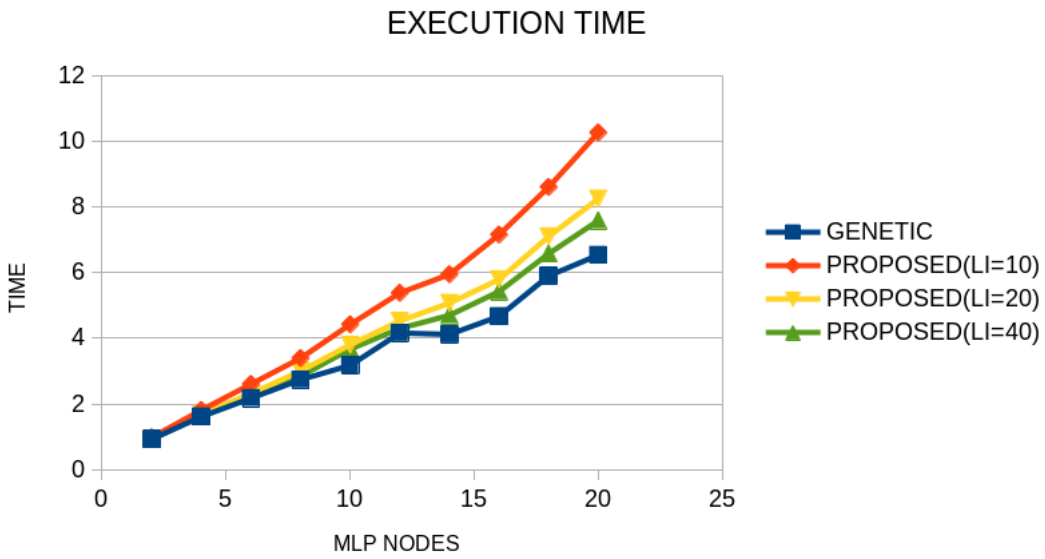

Figure 4. Comparison of execution times for the WINE dataset between the simple Genetic algorithm and the proposed method for various values of the parameter N_I.

3.3. Experiments with the differential weight

An additional experiment was performed to demonstrate the reliability of the proposed methodology. In this experiment, three different differential weight F calculation techniques were used for the proposed operator. The range of this parameter was defined as $[0, 2]$ in the work of Storn and Price [97]. These techniques are the following:

1. Fixed, which is the default technique in the proposed method. In this technique the value $F = 0.8$ is used for the differential weight.
2. Adaptive, where the adaptive calculation of parameter F is uses as proposed in [98].
3. Random, where the stochastic calculation of parameter F as proposed in [99] is used.

In table 4 the results from the application of the proposed method using the previously mentioned techniques for the differential weight are depicted for the classification datasets. Similarly, the same method is applied on the regression datasets and the results are presented in Table 5.

Table 4. Experiments for classification datasets using a series of differential weight mechanisms.

DATASET	FIXED	ADAPTIVE	RANDOM
Alcohol	24.79%	25.65%	23.02%
Appendicitis	15.97%	15.83%	15.80%
Australian	31.76%	31.61%	31.60%
Balance	8.39%	8.45%	8.65%
Circular	3.69%	3.67%	3.78%
Cleveland	48.10%	48.39%	48.35%
Dermatology	7.74%	7.27%	7.37%
Ecoli	47.62%	48.34%	47.96%
Fert	22.00%	22.17%	22.10%
Haberman	25.99%	26.20%	26.12%
Hayes Roth	37.00%	38.38%	37.65%
Heart	24.79%	24.15%	25.51%
HouseVotes	5.22%	5.21%	4.78%
Ionosphere	9.56%	9.28%	9.30%
Liverdisorder	31.08%	31.46%	31.11%
Lymography	28.60%	28.95%	27.88%
Mammographic	16.98%	17.02%	17.18%
Parkinsons	18.02%	17.86%	17.90%
Pima	30.44%	31.12%	30.48%
Popfailures	4.29%	4.25%	4.28%
Regions2	26.43%	25.94%	26.35%
Saheart	32.60%	33.11%	32.92%
Segment	30.00%	28.83%	30.85%
Sonar	18.78%	18.08%	18.70%
Spiral	44.20%	44.21%	44.12%
Statheart	22.72%	22.72%	23.41%
Student	4.16%	3.95%	4.35%
Wdbc	7.73%	7.48%	7.40%
Wine	8.55%	6.29%	6.74%
Z_F_S	6.46%	6.92%	6.65%
ZO_NF_S	6.01%	6.10%	5.89%
ZONF_S	1.79%	1.71%	1.76%
ZOO	9.07%	6.57%	8.90%
AVERAGE	**20.02%**	**19.91%**	**19.97%**

Table 5. Experiments for regression datasets using a variety of weight mechanism methods.

DATASET	FIXED	ADAPTIVE	RANDOM
ABALONE	4.33	4.24	4.34
AIRFOIL	0.003	0.003	0.003
BASEBALL	67.45	67.23	66.76
BK	0.02	0.02	0.02
BL	0.002	0.002	0.002
CONCRETE	0.003	0.003	0.003
DEE	0.20	0.20	0.20
HOUSING	26.62	26.07	26.11
FRIEDMAN	1.33	1.21	1.34
FY	0.039	0.039	0.039
HO	0.014	0.014	0.014
LASER	0.0027	0.0027	0.0028
LW	0.016	0.011	0.011
MB	0.048	0.048	0.048
MORTGAGE	0.31	0.35	0.34
PLASTIC	2.20	2.13	2.20
PL	0.023	0.022	0.022
PY	0.016	0.017	0.017
QUAKE	0.043	0.082	0.04
SN	0.024	0.024	0.024
STOCK	3.47	3.33	3.45
TREASURY	0.44	0.42	0.45
VE	0.023	0.023	0.023
AVERAGE	**4.64**	**4.59**	**4.59**

Furthermore, the statistical comparison for these experimental tables is outlined in Figures 5 and 6 respectively.

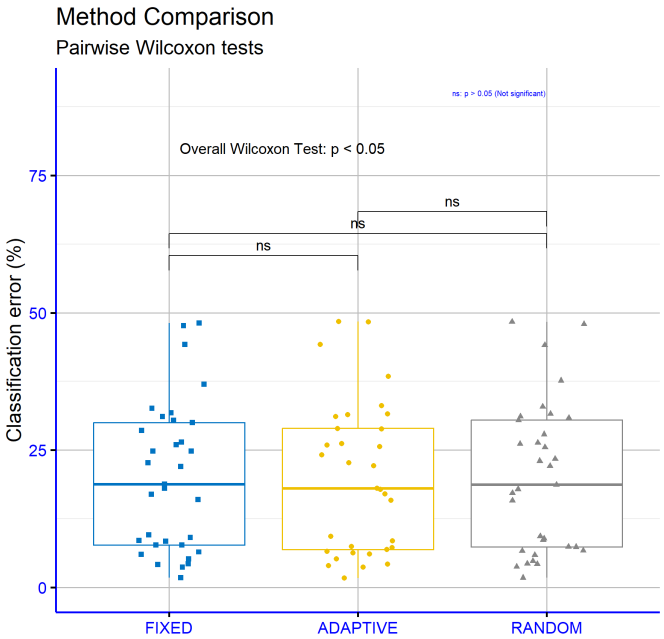

Figure 5. Statistical comparison for the experimental results on the classification datasets, using the proposed method and a series of differential weight techniques.

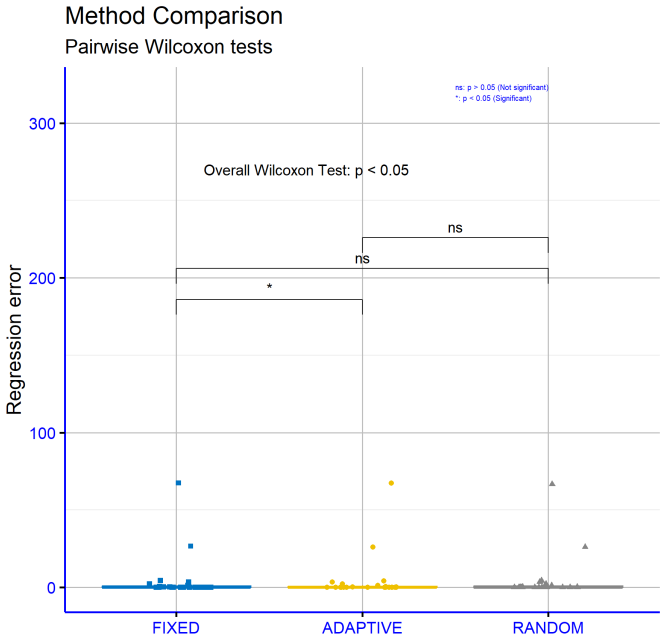

Figure 6. Statistical comparison for the experiments on the regression datasets using the proposed method and a series of differential weight calculation techniques.

When comparing differential weight calculation methods (FIXED, ADAPTIVE, RANDOM), the ADAPTIVE method shows slightly better average performance (19.91% in classification, 4.59 in regression) compared to FIXED (20.02%, 4.64) and RANDOM (19.97%, 4.59). However, the differences are minimal and often statistically insignificant (e.g., $p = 0.83$for FIXED vs ADAPTIVE in classification). In some datasets, such as Lymography and HouseVotes, the RANDOM method outperforms others, highlighting the need to tailor the method to the specific problem. In regression, the only significant difference occurs between FIXED and ADAPTIVE ($p = 0.041$).

For this experiment as shown in Figure 5, the differences between FIXED and ADAPTIVE ($p = 0.83$) and ADAPTIVE vs RANDOM ($p = 0.94$) are not statistically significant, suggesting equivalent performance. However, the comparison FIXED vs RANDOM ($p = 2.3 \times 10^{-10}$) reveals a highly significant difference, likely due to the random nature of RANDOM. For regression datasets (Figure 6), the only significant difference occurs between FIXED and ADAPTIVE ($p = 0.041$), while the remaining comparisons (FIXED vs RANDOM: $p = 0.75$, ADAPTIVE vs RANDOM: $p = 0.31$) lack statistical significance.

3.4. Experiments with the margin factor a

An additional experiment was executed to outline the performance and the stability of the proposed method. In this experiment that margin factor denoted as a in the proposed method was varied from $a = 1.0$ (which is the default value) to $a = 4.0$. The experimental results for the classification datasets are shown in Table 6 and for regression datasets in Table 7.

Table 6. Experiments for classification datasets using a series of values for the margin factor a.

DATASET	$a = 1.0$	$a = 1.5$	$a = 2.0$	$a = 2.5$	$a = 3.0$	$a = 3.5$	$a = 4.0$
Alcohol	24.79%	22.88%	25.55%	30.21%	30.77%	32.05%	28.42%
Appendicitis	15.97%	16.33%	18.07%	18.70%	17.30%	18.60%	19.51%
Australian	31.76%	32.79%	33.23%	32.65%	32.87%	32.76%	33.68%
Balance	8.39%	8.50%	8.39%	8.84%	8.85%	8.87%	8.78%
Circular	3.69%	4.19%	4.22%	4.30%	4.33%	4.46%	4.70%
Cleveland	48.10%	47.17%	46.68%	47.58%	50.84%	50.62%	47.74%
Dermatology	7.74%	7.47%	7.60%	7.57%	7.71%	8.86%	7.83%
Ecoli	47.62%	48.69%	48.29%	49.50%	51.39%	51.99%	49.39%
Fert	22.00%	23.47%	23.50%	25.03%	24.80%	24.17%	24.27%
Haberman	25.99%	26.26%	26.74%	27.23%	27.49%	27.67%	27.14%
Hayes Roth	37.00%	39.15%	39.69%	35.49%	38.10%	39.15%	40.05%
Heart	24.79%	24.64%	24.85%	25.50%	26.85%	26.46%	24.89%
HouseVotes	5.22%	4.91%	5.05%	4.90%	4.57%	4.71%	5.02%
Ionosphere	9.56%	9.65%	9.92%	9.72%	9.75%	9.57%	10.23%
Liverdisorder	31.08%	31.94%	31.42%	30.89%	31.54%	30.92%	32.13%
Lymography	28.60%	28.24%	28.79%	30.21%	29.64%	30.71%	26.69%
Mammographic	16.98%	16.34%	16.35%	15.86%	15.88%	16.35%	16.69%
Parkinsons	18.02%	18.88%	18.56%	17.98%	17.74%	17.95%	19.40%
Pima	30.44%	30.89%	31.11%	32.81%	32.88%	32.83%	31.03%
Popfailures	4.29%	5.07%	5.53%	5.51%	5.54%	5.97%	5.97%
Regions2	26.43%	26.53%	26.30%	26.52%	26.28%	26.26%	25.35%
Saheart	32.60%	31.43%	32.98%	33.52%	33.49%	33.46%	32.93%
Segment	30.00%	27.98%	30.86%	31.39%	33.76%	34.51%	35.41%
Sonar	18.78%	21.08%	22.23%	21.47%	21.75%	21.57%	21.68%
Spiral	44.20%	44.65%	44.52%	43.81%	43.58%	43.39%	44.13%
Statheart	22.72%	23.40%	23.85%	24.22%	25.56%	26.10%	24.22%
Student	4.16%	4.75%	5.24%	5.38%	5.43%	5.55%	5.37%
Wdbc	7.73%	6.95%	6.64%	7.14%	7.29%	7.17%	7.31%
Wine	8.55%	6.59%	9.35%	9.47%	8.12%	9.65%	11.02%
Z_F_S	6.46%	6.66%	6.90%	6.92%	6.53%	6.87%	6.61%
ZO_NF_S	6.01%	6.03%	6.08%	6.95%	7.17%	7.28%	6.25%
ZONF_S	1.79%	1.75%	1.80%	2.15%	2.14%	2.13%	1.79%
ZOO	9.07%	8.63%	7.00%	7.87%	6.77%	6.97%	7.53%
AVERAGE	**20.02%**	**20.12%**	**20.52%**	**20.83%**	**21.11%**	**21.38%**	**21.00%**

Table 7. Experiments for regression datasets using a variety of values for the margin factor a.

DATASET	$a = 1.0$	$a = 1.5$	$a = 2.0$	$a = 2.5$	$a = 3.0$	$a = 3.5$	$a = 4.0$
ABALONE	4.33	4.39	4.46	4.64	4.84	4.93	5.06
AIRFOIL	0.003	0.003	0.002	0.002	0.003	0.003	0.002
BASEBALL	67.45	74.66	79.78	79.39	81.55	84.65	86.54
BK	0.02	0.019	0.019	0.018	0.017	0.018	0.02
BL	0.002	0.001	0.001	0.0007	0.0009	0.0008	0.003
CONCRETE	0.003	0.003	0.003	0.004	0.003	0.004	0.003
DEE	0.20	0.20	0.20	0.21	0.20	0.21	0.20
HOUSING	26.62	26.24	28.13	27.91	27.36	29.53	29.25
FRIEDMAN	1.33	1.19	1.20	1.21	1.22	1.21	1.20
FY	0.039	0.04	0.042	0.041	0.042	0.042	0.045
HO	0.014	0.014	0.013	0.013	0.013	0.014	0.014
LASER	0.0027	0.0025	0.0025	0.0028	0.0026	0.0025	0.0024
LW	0.016	0.011	0.011	0.013	0.013	0.014	0.013
MB	0.048	0.049	0.051	0.051	0.052	0.054	0.09
MORTGAGE	0.31	0.21	0.37	0.48	0.63	0.68	0.62
PLASTIC	2.20	2.05	2.18	2.23	2.25	2.23	2.21
PL	0.023	0.023	0.021	0.022	0.022	0.022	0.022
PY	0.016	0.022	0.023	0.026	0.028	0.027	0.028
QUAKE	0.043	0.038	0.04	0.039	0.037	0.037	0.039
SN	0.024	0.024	0.025	0.024	0.026	0.024	0.026
STOCK	3.47	3.59	3.68	3.70	3.37	3.24	3.35
TREASURY	0.44	0.42	0.40	0.65	0.82	0.95	1.01
VE	0.023	0.024	0.024	0.025	0.026	0.027	0.029
AVERAGE	**4.64**	**4.92**	**5.25**	**5.25**	**5.33**	**5.56**	**5.64**

Also, the statistical comparisons for these experiments are shown graphically in Figures 7 and 8 respectively.

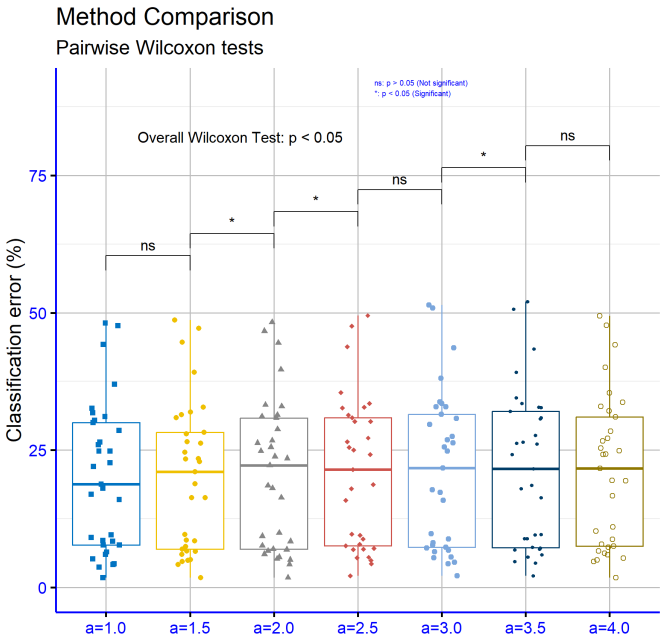

Figure 7. Statistical comparison for the experimental results on the classification datasets, using the proposed method and different values for the margin factor a.

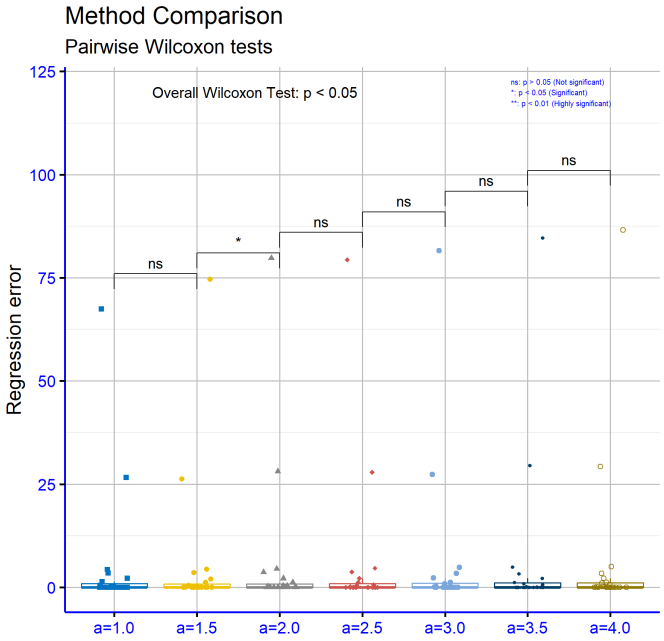

Figure 8. Statistical comparison for the experiments on the regression datasets, using the proposed method and different values for the margin factor a.

The table 6 presents the percentage error values corresponding to various classification datasets for different values of the parameter a which determines the optimal value boundaries in the proposed machine learning model. The last row of the table includes the average error for each value of the parameter a. The data analysis reveals that an increase in the parameter a generally leads to a rise in the average error rate. Specifically, the average increases from 20.02% for $\alpha = 1.0$ to 21.38% for $\alpha = 3.5$, while for $\alpha = 4.0$, it slightly decreases to 21.00%. This indicates that, despite the general upward trend, there are cases where further increasing the parameter can reduce the error. A detailed examination of the data by dataset shows significant variations among different values of the parameter a. For example, in the Circular dataset, the error gradually increases from 3.69% for $\alpha = 1.0$ to 4.70% for $\alpha = 4.0$. Similarly, in the Wine dataset, the error value increases from 8.55% for $\alpha = 1.0$ to 11.02% for $\alpha = 4.0$, indicating a clear negative impact of increasing the parameter. However, in other datasets, the effect of the parameter is neither as linear nor as pronounced. For instance, in the Hayes Roth dataset, the error decreases from 39.69% for $\alpha = 2.0$ to 35.49% for $\alpha = 2.5$, before increasing again. Similarly, in the Lymography dataset, the error significantly decreases for $\alpha = 4.0$ (26.69%) compared to the other parameter values. These data demonstrate that the relationship between the parameter a and the error is not linear across all datasets and depends on the specific characteristics of each dataset. Overall, the statistical analysis indicates that while increasing the parameter a is often associated with higher errors, there are exceptions that may be linked to specific properties of the data or the model.

The table 7 presents the absolute error values for various regression datasets in relation to different values of the parameter a which determines the optimal value boundaries in the proposed machine learning model. The last row lists the average errors for each value of the parameter a. The table analysis shows that an increase in the parameter a is accompanied by a general increase in the average error. Specifically, the average error increases from 4.64 for $\alpha = 1.0$ to 5.64 for $\alpha = 4.0$, suggesting that higher parameter values may lead to worse model performance. Examining the individual datasets reveals variability in the effect of the parameter a. For example, in the BASEBALL dataset, the error continuously increases from 67.45 for $\alpha = 1.0$ to 86.54 for $\alpha = 4.0$, indicating a clear negative effect of increasing the parameter. Conversely, in the STOCK dataset, an increase in the parameter is accompanied

by a decrease in error, from 3.47 for $\alpha = 1.0$ to 3.35 for $\alpha = 4.0$. In other datasets, the effect is less straightforward. For example, in the MORTGAGE dataset, the error increases from 0.31 for $\alpha = 1.0$ to 0.68 for $\alpha = 3.5$ but slightly decreases to 0.62 for $\alpha = 4.0$. Overall, the statistical analysis suggests that the effect of the parameter a depends on the characteristics of each dataset and that, although increasing the parameter is often associated with higher errors, there are cases where performance remains stable or improves slightly.

In Figure 7, the results pertain to classification datasets and various models. From the comparisons between different values of the parameter a it is observed that the comparison between $a = 1.0$ and $a = 1.5$ does not exhibit a statistically significant difference ($p = 0.38$). In contrast, the comparison between $a = 1.5$ and $a = 2.0$ shows a significant difference ($p = 0.018$), as does the comparison between $a = 2.0$ and $a = 2.5$ ($p = 0.045$). The remaining comparisons, specifically between $a = 2.5$ and $a = 3.0$ ($p = 0.25$), $a = 3.0$ and $a = 3.5$ ($p = 0.27$), and $a = 3.5$ and $a = 4.0$ ($p = 0.39$), do not demonstrate statistically significant differences.

In Figure 8, the data refer to regression datasets and comparisons for various models. The results indicate that the comparison between $a = 1.0$ and $a = 1.5$ is not statistically significant ($p = 0.55$), whereas the difference between $a = 1.5$ and $a = 2.0$ is statistically significant ($p = 0.014$). The comparison between $a = 2.0$ and $a = 2.5$ does not show statistical significance ($p = 0.17$), nor do the comparisons between $a = 2.5$ and $a = 3.0$ ($p = 0.2$), $a = 3.0$ and $a = 3.5$ ($p = 0.16$), and $a = 3.5$ and $a = 4.0$ ($p = 0.31$). Overall, the p values highlight significant differences in specific cases, while in others, the differences remain statistically insignificant.

4. Conclusions

The study confirms the superiority of the proposed model compared to existing methods in both classification and regression tasks. This superiority is demonstrated by a statistically significant reduction in error, although the model's performance heavily depends on data characteristics such as complexity, stratification, or the presence of noise. This highlights the importance of adapting the model to the specificities of each problem, as overly generalized approaches may lead to suboptimal results. Additionally, the choice of methods for calculating critical parameters, such as differential weights, appears to have a relatively limited impact on overall performance. Differences between approaches (e.g., FIXED, ADAPTIVE, RANDOM) are minimal and often statistically insignificant, suggesting that optimization efforts could focus on other factors. Tuning the parameter "a," however, emerges as a critical factor. Higher values of this parameter often improve accuracy, particularly for large or complex datasets, but they may also introduce instability in applications where data sensitivity is high (e.g., financial forecasting). This dual behavior underscores the need to balance flexibility and robustness during the tuning process. Despite its analytical approach, the study does not sufficiently explore the potential real-world applications of the proposed model beyond benchmark datasets. Addressing this gap, future research could expand the methodology to practical domains such as biomedical data analysis, social networks, or dynamic systems, including real-time monitoring and financial forecasting. Investigating the model's adaptability and scalability in these fields could enhance its utility in real-world scenarios.

For future research, it would be beneficial to develop dynamic algorithms that automatically adjust the parameter a based on data dynamics. For example, machine learning mechanisms that analyze data variability or heterogeneity in real-time could optimize the value of a without human intervention, ensuring both stability and high performance. Additionally, the development of hybrid methods that combine the adaptability of ADAPTIVE approaches with the simplicity of FIXED approaches could result in more resilient solutions capable of addressing a broader range of problems. It is also important to investigate how specific data characteristics—such as noisy measurements, class imbalances, or lack of labeling—affect the model's effectiveness. Such analysis would support the development of adaptive strategies for data preprocessing or augmentation. Furthermore, exploring the

generalizability of results to more complex environments, such as time-series data or dynamic systems, remains an open research avenue. Finally, integrating explainable artificial intelligence (XAI) techniques, such as feature contribution analysis or visualization tools, could enhance the model's transparency. This would not only facilitate the interpretation of results by experts but also aid in identifying optimal parameter-tuning practices, making the model a more predictable and reliable tool for real-world applications.

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.

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 conflict of interest.

Sample Availability: Not applicable.

References

1. Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., & Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. Heliyon, 4(11).
2. Suryadevara, S., & Yanamala, A. K. Y. (2021). A Comprehensive Overview of Artificial Neural Networks: Evolution, Architectures, and Applications. Revista de Inteligencia Artificial en Medicina, 12(1), 51-76.
3. P. Baldi, K. Cranmer, T. Faucett et al, Parameterized neural networks for high-energy physics, Eur. Phys. J. C **76**, 2016.
4. Baldi, P., Cranmer, K., Faucett, T., Sadowski, P., & Whiteson, D. (2016). Parameterized neural networks for high-energy physics. The European Physical Journal C, 76(5), 1-7.
5. Firth, A. E., Lahav, O., & Somerville, R. S. (2003). Estimating photometric redshifts with artificial neural networks. Monthly Notices of the Royal Astronomical Society, 339(4), 1195-1202.
6. Shen, Y., & Bax, A. (2013). Protein backbone and sidechain torsion angles predicted from NMR chemical shifts using artificial neural networks. Journal of biomolecular NMR, 56, 227-241.
7. Z. Huang, H. Chen, C.-Jung Hsu, W.-Hwa Chen, S. Wu, Credit rating analysis with support vector machines and neural networks: a market comparative study, Decision Support Systems **37**, pp. 543-558, 2004.
8. Igor I. Baskin, David Winkler and Igor V. Tetko, A renaissance of neural networks in drug discovery, Expert Opinion on Drug Discovery **11**, pp. 785-795, 2016.
9. Ronadl Bartzatt, Prediction of Novel Anti-Ebola Virus Compounds Utilizing Artificial Neural Network (ANN), Chemistry Faculty Publications **49**, pp. 16-34, 2018.
10. D.E. Rumelhart, G.E. Hinton and R.J. Williams, Learning representations by back-propagating errors, Nature **323**, pp. 533 - 536 , 1986.
11. T. Chen and S. Zhong, Privacy-Preserving Backpropagation Neural Network Learning, IEEE Transactions on Neural Networks **20**, , pp. 1554-1564, 2009.
12. M. Riedmiller and H. Braun, A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP algorithm, Proc. of the IEEE Intl. Conf. on Neural Networks, San Francisco, CA, pp. 586–591, 1993.
13. T. Pajchrowski, K. Zawirski and K. Nowopolski, Neural Speed Controller Trained Online by Means of Modified RPROP Algorithm, IEEE Transactions on Industrial Informatics **11**, pp. 560-568, 2015.
14. Rinda Parama Satya Hermanto, Suharjito, Diana, Ariadi Nugroho, Waiting-Time Estimation in Bank Customer Queues using RPROP Neural Networks, Procedia Computer Science **135**, pp. 35-42, 2018.
15. B. Robitaille and B. Marcos and M. Veillette and G. Payre, Modified quasi-Newton methods for training neural networks, Computers & Chemical Engineering **20**, pp. 1133-1140, 1996.
16. Q. Liu, J. Liu, R. Sang, J. Li, T. Zhang and Q. Zhang, Fast Neural Network Training on FPGA Using Quasi-Newton Optimization Method,IEEE Transactions on Very Large Scale Integration (VLSI) Systems **26**, pp. 1575-1579, 2018.
17. C.L. Kuo, E.E. Kuruoglu, W.K.V. Chan, Neural Network Structure Optimization by Simulated Annealing, Entropy **24**, 348, 2022.

18. C. Zhang, H. Shao and Y. Li, Particle swarm optimisation for evolving artificial neural network, IEEE International Conference on Systems, Man, and Cybernetics, , pp. 2487-2490, 2000.
19. Jianbo Yu, Shijin Wang, Lifeng Xi, Evolving artificial neural networks using an improved PSO and DPSO **71**, pp. 1054-1060, 2008.
20. F. H. F. Leung, H. K. Lam, S. H. Ling and P. K. S. Tam, Tuning of the structure and parameters of a neural network using an improved genetic algorithm, IEEE Transactions on Neural Networks **14**, pp. 79-88, 2003
21. X. Yao, Evolving artificial neural networks, Proceedings of the IEEE, 87(9), pp. 1423-1447, 1999.
22. Wang, L., Zeng, Y., & Chen, T. (2015). Back propagation neural network with adaptive differential evolution algorithm for time series forecasting. Expert Systems with Applications, 42(2), 855-863.
23. K.M. Salama, A.M. Abdelbar, Learning neural network structures with ant colony algorithms, Swarm Intell **9**, pp. 229–265, 2015.
24. S. Mirjalili, How effective is the Grey Wolf optimizer in training multi-layer perceptrons, Appl Intell **43**, pp. 150–161, 2015.
25. I. Aljarah, H. Faris, S. Mirjalili, Optimizing connection weights in neural networks using the whale optimization algorithm, Soft Comput **22**, pp. 1–15, 2018.
26. J.-R. Zhang, J. Zhang, T.-M. Lok, M.R. Lyu, A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training, Applied Mathematics and Computation **185**, pp. 1026-1037, 2007.
27. K-Su Oh, K. Jung, GPU implementation of neural networks, Pattern Recognition **37**, pp. 1311-1314, 2004.
28. M. Zhang, K. Hibi, J. Inoue, GPU-accelerated artificial neural network potential for molecular dynamics simulation, Computer Physics Communications **285**, 108655, 2023.
29. Krüger, T., Kusumaatmaja, H., Kuzmin, A., Shardt, O., Silva, G., & Viggen, E. M. (2017). The lattice Boltzmann method. Springer International Publishing, 10(978-3), 4-15.
30. Kang, Q., Li, K. Q., Fu, J. L., & Liu, Y. (2024). Hybrid LBM and machine learning algorithms for permeability prediction of porous media: a comparative study. Computers and Geotechnics, 168, 106163.
31. I. Ivanova, M. Kubat, Initialization of neural networks by means of decision trees, Knowledge-Based Systems **8**, pp. 333-344, 1995.
32. J.Y.F. Yam, T.W.S. Chow, A weight initialization method for improving training speed in feedforward neural network, Neurocomputing **30**, pp. 219-232, 2000.
33. K. Chumachenko, A. Iosifidis, M. Gabbouj, Feedforward neural networks initialization based on discriminant learning, Neural Networks **146**, pp. 220-229, 2022.
34. T.M. Varnava, A.J.Meade, An initialization method for feedforward artificial neural networks using polynomial bases, Advances in Adaptive Data Analysis **3**, pp. 385-400, 2011.
35. S.S. Sodhi, P. Chandra, Interval based Weight Initialization Method for Sigmoidal Feedforward Artificial Neural Networks, AASRI Procedia **6**, pp. 19-25, 2014.
36. Narkhede, M. V., Bartakke, P. P., & Sutaone, M. S. (2022). A review on weight initialization strategies for neural networks. Artificial intelligence review, 55(1), 291-322.
37. J. Arifovic, R. Gençay, Using genetic algorithms to select architecture of a feedforward artificial neural network, Physica A: Statistical Mechanics and its Applications **289**, pp. 574-594, 2001.
38. P.G. Benardos, G.C. Vosniakos, Optimizing feedforward artificial neural network architecture, Engineering Applications of Artificial Intelligence **20**, pp. 365-382, 2007.
39. B.A. Garro, R.A. Vázquez, Designing Artificial Neural Networks Using Particle Swarm Optimization Algorithms, Computational Intelligence and Neuroscience, 369298, 2015.
40. B. Baker, O. Gupta, N. Naik, R. Raskar, Designing neural network architectures using reinforcement learning. arXiv preprint arXiv:1611.02167, 2016.
41. M. O'Neill, C. Ryan, Grammatical evolution, IEEE Trans. Evol. Comput. **5,**pp. 349–358, 2001.
42. I.G. Tsoulos, D. Gavrilis, E. Glavas, Neural network construction and training using grammatical evolution, Neurocomputing **72**, pp. 269-277, 2008.
43. Pant, M., Zaheer, H., Garcia-Hernandez, L., & Abraham, A. (2020). Differential Evolution: A review of more than two decades of research. Engineering Applications of Artificial Intelligence, 90, 103479.
44. Li, Y. H., Wang, J. Q., Wang, X. J., Zhao, Y. L., Lu, X. H., & Liu, D. L. (2017). Community detection based on differential evolution using social spider optimization. Symmetry, 9(9), 183.
45. Yang, W., Siriwardane, E. M. D., Dong, R., Li, Y., & Hu, J. (2021). Crystal structure prediction of materials with high symmetry using differential evolution. Journal of Physics: Condensed Matter, 33(45), 455902.
46. Lee, C. Y., & Hung, C. H. (2021). Feature ranking and differential evolution for feature selection in brushless DC motor fault diagnosis. Symmetry, 13(7), 1291.
47. Saha, S., & Das, R. (2018). Exploring differential evolution and particle swarm optimization to develop some symmetry-based automatic clustering techniques: application to gene clustering. Neural Computing and Applications, 30, 735-757.
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. Prasad, T., Park, N., Multiobjective Genetic Algorithms for Design of Water Distribution Networks, J. Water Resour. Plann. Manage. **130**, pp. 73-82, 2004.
50. Sung-Hwan Min, Jumin Lee, Ingoo Han, Hybrid genetic algorithms and support vector machines for bankruptcy prediction, Expert Systems with Applications **31**, pp. 652-660, 2006.

51. D. Whitley, T. Starkweather, C. Bogart, Genetic algorithms and neural networks: Optimizing connections and connectivity, Parallel Computing **14**, pp. 347-361, 1990.
52. P. Kaelo, M.M. Ali, Integrated crossover rules in real coded genetic algorithms, European Journal of Operational Research **176**, pp. 60-76, 2007.
53. M. Kelly, R. Longjohn, K. Nottingham, The UCI Machine Learning Repository. 2023. Available online: https://archive.ics.uci.edu (accessed on 20 September 2023).
54. 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.
55. 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.
56. 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.
57. J.R. Quinlan, Simplifying Decision Trees. International Journal of Man-Machine Studies **27**, pp. 221-234, 1987.
58. T. Shultz, D. Mareschal, W. Schmidt, Modeling Cognitive Development on Balance Scale Phenomena, Machine Learning **16**, pp. 59-88, 1994.
59. 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.
60. R. Setiono , W.K. Leow, FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks, Applied Intelligence **12**, pp. 15-25, 2000.
61. 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.
62. 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.
63. 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.
64. I. Kononenko, E. Šimec, M. Robnik-Šikonja, Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF, Applied Intelligence **7**, pp. 39–55, 1997.
65. 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.
66. J.G. Dy , C.E. Brodley, Feature Selection for Unsupervised Learning, The Journal of Machine Learning Research **5**, pp 845–889, 2004.
67. S. J. Perantonis, V. Virvilis, Input Feature Extraction for Multilayered Perceptrons Using Supervised Principal Component Analysis, Neural Processing Letters **10**, pp 243–252, 1999.
68. J. Garcke, M. Griebel, Classification with sparse grids using simplicial basis functions, Intell. Data Anal. **6**, pp. 483-502, 2002.
69. J. Mcdermott, R.S. Forsyth, Diagnosing a disorder in a classification benchmark, Pattern Recognition Letters **73**, pp. 41-43, 2016.
70. 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.
71. 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.
72. 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.
73. 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.
74. 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.
75. 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.
76. 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.
77. T. Hastie, R. Tibshirani, Non-parametric logistic and proportional odds regression, JRSS-C (Applied Statistics) **36**, pp. 260–276, 1987.
78. M. Dash, H. Liu, P. Scheuermann, K. L. Tan, Fast hierarchical clustering and its validation, Data & Knowledge Engineering **44**, pp 109–138, 2003.
79. Gorman, R.P.; Sejnowski, T.J. Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets. Neural Netw. 1988, 1, 75–89.

80. 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.
81. W.H. Wolberg, O.L. Mangasarian, Multisurface method of pattern separation for medical diagnosis applied to breast cytology, Proc Natl Acad Sci U S A. **87**, pp. 9193–9196, 1990.
82. 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.
83. P. Zhong, M. Fukushima, Regularized nonsmooth Newton method for multi-class support vector machines, Optimization Methods and Software **22**, pp. 225-236, 2007.
84. 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.
85. 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.
86. M. Koivisto, K. Sood, Exact Bayesian Structure Discovery in Bayesian Networks, The Journal of Machine Learning Research **5**, pp. 549–573, 2004.
87. 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
88. 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 5 March 2025).
89. J.S. Simonoff, Smooting Methods in Statistics, Springer - Verlag, 1996.
90. I.Cheng Yeh, Modeling of strength of high performance concrete using artificial neural networks, Cement and Concrete Research. **28**, pp. 1797-1808, 1998.
91. D. Harrison and D.L. Rubinfeld, Hedonic prices and the demand for clean ai, J. Environ. Economics & Management **5**, pp. 81-102, 1978.
92. Friedman, J. (1991): Multivariate Adaptative Regression Splines. Annals of Statistics, 19:1, 1--141.
93. R.D. King, S. Muggleton, R. Lewis, M.J.E. Sternberg, Proc. Nat. Acad. Sci. USA **89**, pp. 11322–11326, 1992.
94. 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.
95. M.J.D Powell, A Tolerant Algorithm for Linearly Constrained Optimization Calculations, Mathematical Programming **45**, pp. 547-566, 1989.
96. K. O. Stanley, R. Miikkulainen, Evolving Neural Networks through Augmenting Topologies, Evolutionary Computation **10**, pp. 99-127, 2002.
97. Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, 11, 341-359.
98. Wu, K., Liu, Z., Ma, N., & Wang, D. (2022). A Dynamic Adaptive Weighted Differential Evolutionary Algorithm. Computational Intelligence and Neuroscience, 2022(1), 1318044.
99. Charilogis V, Tsoulos IG, Tzallas A, Karvounis E. Modifications for the Differential Evolution Algorithm. Symmetry. 2022; 14(3):447.