Article

Optimizing the bounds of neural networks using a novel simulated annealing method

Ioannis G. Tsoulos[1,*], Vasileios Charilogis[2] and 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 reliable machine learning models that have been applied to a multitude of practical and scientific applications in recent decades. Among these applications there are examples from the areas of physics, chemistry, medicine, etc. For their effective application to these problems, it is necessary to adapt their parameters using optimization techniques. However, in order to be effective, optimization techniques must know the range of values for the parameters of the artificial neural network, so that they can adequately train the artificial neural network. In most cases, this is not possible, as these ranges are also significantly affected by the inputs to the artificial neural network from the objective problem it is called upon to solve. This situation usually results in artificial neural networks becoming trapped in local minima of the error function or, even worse, in the phenomenon of overfitting, where although the training error achieves low values, the artificial neural network exhibits low performance in the corresponding test set. To address this limitation, this work proposes a novel two-stage training approach in which a simulated annealing (SA)–based preprocessing stage is employed to automatically identify optimal parameter value intervals before the application of any optimization method to train the neural network. Unlike similar approaches that rely on fixed or heuristically selected parameter bounds, the proposed pre-processing technique explores the parameter space probabilistically, guided by a temperature-controlled acceptance mechanism that balances global exploration and local refinement. The proposed method has been successfully applied to a wide range of classification and regression problems and comparative results are presented in detail in the present work.

Keywords: Neural networks; Simulated Annealing; Optimization methods

Received:
Revised:
Accepted:
Published:

1. Introduction

Artificial neural networks are among the most commonly used machine learning models for solving classification and regression tasks [1,2]. These models are typically formulated as functions $N(\overrightarrow{x},\overrightarrow{w})$, where the vector $\overrightarrow{x} \in \mathbb{R}^d$ represents the input pattern and $\overrightarrow{w}$ denotes the set of trainable parameters of the neural network. Model training is achieved by minimizing the training error, which is defined by the following expression:

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

The set $\left(\overrightarrow{x_i}, y_i\right),\ i = 1, ..., M$ stands for the associated training set of the objective problem. In this set each value y_i corresponds to the desired output for the pattern $\overrightarrow{x_i}$. As suggested in [3], a neural network could be expressed as a function with the following definition:

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

In this equation, the value H represents the number of desired processing units. Also, the function $\sigma(x)$ stands for the the sigmoid function, expressed as:

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

The total number of parameters for neural network can be calculated as $n = (d+2)H$, using the equation 2. Moreover, a series of alternative activation functions has been proposed in the related literature, such as the tanh expressed with equation 4.

$$\tanh(x) = \frac{e^{2x}+1}{e^{2x}-1} \quad (4)$$

Additionally, Guarnieri et al proposed the incorporation of an adaptive spline function as the activation function[4]. Likewise, Ertuğrul proposed the usage of a trained activation function [5]. A systematic review on activation functions for neural networks can be found in the recent publication of Rasamoelina et al [6].

These machine learning models have been used efficiently in a series of practical problems, such as image processing [7], time series forecasting [8], economic problems [9], problems presented in experiments in physics [10,11] etc. More recent works applied the neural networks to problems appeared in flood simulation [12], solar radiation prediction [13], agricultural [14], computer networks [15], medicine [16,17], mechanical applications [18] etc.

The equation 1 has been tackled by a variety of optimization procedures during the past years, such as the Back Propagation algorithm [19], the RPROP algorithm [20,21], the ADAM method [22] etc. Additionally, global optimization methods have been incorporated to train neural networks. Among them one can detect the usage of Genetic Algorithms [23,24], the Particle Swarm Optimization (PSO) method [25], the Simulated Annealing optimization technique [26], the Differential Evolution method [27], the Artificial Bee Colony (ABC) method [28], application of the Gray Wolf optimization method [29], etc. Also, Sexton et al introduced the usage of the tabu search algorithm in the training process of neural networks [30]. Also, a hybrid algorithm that incorporates the PSO method and the Back Propagation algorithm was suggested by Zhang et al [31]. Zhao et al introduced a Cascaded Forward Algorithm for optimal training of artificial neural networks [32]. Moreover, the extensive adoption of parallel computing approaches in recent years has led to numerous studies focusing on the training of artificial neural networks using these methods [33,34].

Furthermore, Artificial Intelligence (AI) has been incorporated successfully in metahumanistics, a transdisciplinary field that examines the co-evolution of human cognition, culture, and technology beyond traditional human-centered frameworks. Machine learning models capable of language generation, image synthesis, and pattern discovery challenge classical distinctions between human and machine authorship, interpretation, and agency [35,36].

Although, the previously mentioned method face a series of numeric problems, such as trapping in local minima of the error function provided in equation 1 or sometimes are prone to the issue of overifitting. In overfitting, the neural network demonstrates degraded performance when applied to data that were not present during the training phase. The overfitting issue has been studied by various researches during the past years and a variety of algorithms have been proposed for this problem, such as the weight sharing method [37,38], pruning methods [39,40], early stopping techniques [41,42], weight decaying [43,44] etc. Another way to handle this problem is to dynamically create the architecture of artificial neural networks. In this direction of research, various studies were presented, such as the use of genetic algorithms [45,46] or the use of the PSO technique for the efficient creation of the architecture of artificial neural networks [47]. Another method that was proposed to handle the overfitting problem is the incorporation of reinforcement learning as suggested by Siebel at al. [48].

The current work introduces a new method, that is based on the Simulated Annealing approach to identify the optimal range of values for the parameters of the neural network before the incorporation of the training method. The first phase method creates various test value intervals which it adjusts appropriately through the usage of an innovative technique based on Simulated Annealing. This method starts from an initial range of values which can be expanded or contracted in an attempt to avoid the phenomenon of overfitting. After the end of the first phase, the artificial neural network is trained within the optimal value interval identified in the first phase using any optimization technique.

Determining appropriate initial parameter constraints is a major issue in neural network training because the choice of initial values strongly influences optimization dynamics, convergence speed, and final model performance. Poor initialization—such as weights that are too large or too small—can lead to numerical and optimization issues like vanishing or exploding gradients, which hinder effective gradient-based learning, especially in deep architectures. As a result, training may become unstable, excessively slow, or fail altogether. There is no method that adaptively optimizes weight constraints based on specific data characteristics before the main training.

There are several initialization methods from the related literature, such as sampling weights from a standard normal or uniform distribution without accounting the size of the neural networks, can lead to vanishing or exploding gradients, particularly in deep architectures, severely impairing convergence [49]. Similarly, the Xavier initialization method defines the variance of the initial weights based on the number of input and output units [50]. While effective for sigmoid and tanh activations, this approach has been shown to perform poorly with rectified linear units. In the same direction of research, He initialization extends the previous method by adapting the variance scaling, improving training stability under specific assumptions regarding activation sparsity [51]. However, its effectiveness remains dependent on the network architecture and activation behavior, and inappropriate parameter constraints can still degrade optimization.

The rest of this manuscript is organized as follows: the used dataset and the incorporated methods used in the conducted experiments are outlined in section 2, the experimental results are shown and discussed in section 3 and finally a detailed discussion is provided in section 4.

2. Materials and Methods

In this section, the new technique for locating the value range for the parameters of the artificial neural network will be presented as well as the final method for training the parameters within the optimal value range of the first phase. In the current work, a genetic algorithm was used for the final adjustment of the parameters of the artificial neural network.

2.1. The proposed Simulated Annealing variant

A variant of the Simulated Annealing procedure is used a the method of the first phase of the current work. The Simulated Annealing is an optimization procedure [52], that can be used in a magnitude of problems, such as resource allocation [53], portfolio problems [54], energy problems [55], biology [56], multi objective optimization [57], the timetabling problem [58], the traveling salesman problem [59] etc. The behavior of this algorithm is controlled by a positive parameter that resembles the annealing temperature in physics. At high temperatures the algorithm performs a wider search in the value range of the objective problem and as the temperature decreases the algorithm focuses on some possible minima of the objective function. Temperature reduction can be achieved by different cooling mechanisms, as reported in the relevant literature [60,61].

The proposed modification constructs candidate intervals for the neural network parameters and assesses each interval by generating multiple random parameter configurations within it. The mean error of the resulting neural networks is then used as the performance measure for the corresponding interval. Also, To avoid the phenomenon of overtraining, a technique proposed in the publication of Anastasopoulos et al. [62] is used and the quantity $B\left(N\left(\overrightarrow{x},\overrightarrow{w}\right),a\right)$ introduced in that publication is used here also. To calculate this quantity, it is assumed that the sigmoid function with the following definition is used as the activation function of the artificial neural network:

$$\sigma(x)=\frac{1}{1+\exp(-x)} \tag{5}$$

A typical plot for the sigmoid function in the range $[-5,5]$ is outlined in Figure 1.

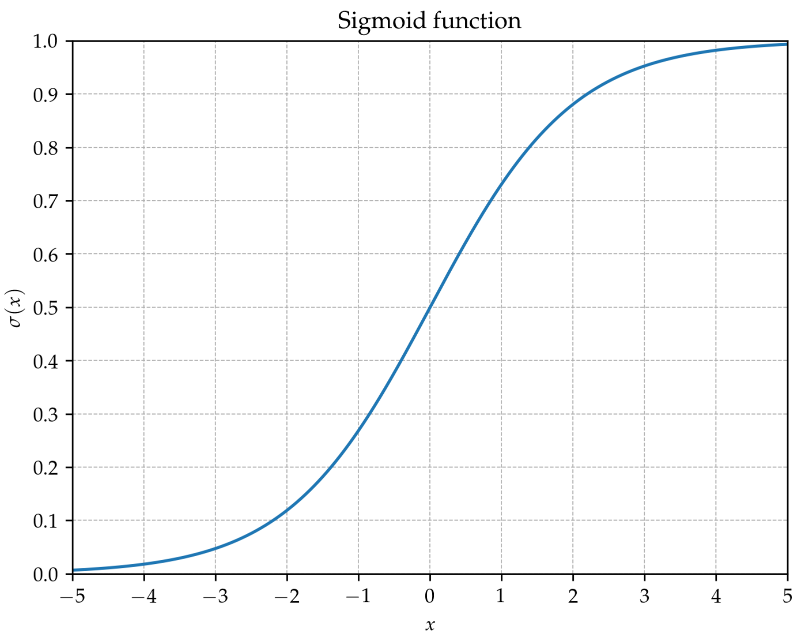

Figure 1. Plot of the sigmoid function $\sigma(x)$ in the range $[-5,5]$.

As can be observed from both the analytical expression and the corresponding figure, the function exhibits rapid saturation, asymptotically approaching 1 for large positive values of the parameter x and converging to 0 for large negative values. This saturation behavior significantly limits the function's generalization capacity, since substantial variations in x result in only marginal changes in the output of the sigmoid function. The quantity $B\left(N\left(\overrightarrow{x},\overrightarrow{w}\right),a\right)$ was introduced in that paper can be used to measure this effect.

This quantity is computed according to Algorithm 1. The resulting function can be employed as a regularization mechanism to mitigate overfitting by constraining the neural network parameters within problem-dependent intervals. The user-defined parameter a serves as an upper bound on the input of the sigmoid unit. When this bound is exceeded, the network is likely to exhibit diminished generalization performance, as the sigmoid function enters a saturation regime in which variations in the input no longer produce meaningful changes in the output.

Algorithm 1 The following algorithm is used to calculate the quantity for neural network $N(x,w)$ and a provided input a.

```
function calculateB$\left(N\left(\overrightarrow{x},\overrightarrow{w}\right),a\right)$
1. Inputs: The Neural network $N\left(\overrightarrow{x},\overrightarrow{w}\right)$ and the bound parameter $a,\ a>1$.
2. Set $k=0$
3. For $i=1..H$ Do
    (a) For $j=1..M$ Do
        i. Set $b=\sum_{k=1}^{d} w_{(d+2)i-(d+i)+k}x_{jk}+w_{(d+2)i}$
        ii. If $|b|>a$ set $k=k+1$
    (b) EndFor
4. EndFor
5. Return $\frac{k}{H\star M}$
End Function
```

As can be observed from both the equation of the sigmoid equation and the corresponding figure, when the parameter x increases, the function rapidly approaches the value 1. Conversely, as x decreases, the function takes values that are very close to 0. This behavior indicates that the function quickly loses its generalization capability, since significant variations in the parameter x no longer lead to proportional changes in the output of the sigmoid function. For this reason, the quantity calculateB$\left(N\left(\overrightarrow{x},\overrightarrow{w}\right),a\right)$ was used to quantify this effect. It can be employed to prevent overfitting by constraining the neural network parameters within intervals that depend on the specific problem at hand. The user-defined parameter $a$serves as an upper bound for the input of the sigmoid unit. When this bound is exceeded, the neural network is likely to exhibit reduced generalization ability, as the sigmoid output remains nearly constant regardless of further changes in the input.

The proposed Simulated Annealing variant produces randomly intervals for the parameters of the neural network. Each interval is evaluated for its efficiency using the procedure presented in Algorithm 3. The behavior of the proposed method is controlled by a variable representing the temperature. At high temperatures the method searches a wider range of the problem and may generate value ranges for the neural network parameters that will have a large range. However, as the temperature drops, the method focuses on value ranges that are more promising. The main steps of this method are given in Algorithm 2.

Algorithm 2 The main steps of the proposed Simulated Annealing variant.

```
Function simanMethod$\left(I_w, T_0, r_T, N_{\text{eps}}, a, p_c\right)$
Inputs:
  • The weight factor $I_w$
  • The initial temperature $T_0$
  • The reduction factor used for the temperature $r_T,\ r_T > 0,\ r_T < 1$
  • The number of random intervals produced at each iteration is denoted as $N_{\text{eps}}$
  • The scale factor $a$ used in the function $B\left(N\left(\overrightarrow{x}, \overrightarrow{w}\right), a\right)$.
  • The perturbation factor $p_c > 0$.
1. Construct the initial bound vectors $L^*,\ R^*$ as follows:
       $L_i^* = -I_w \quad , \quad i = 1, \ldots, n$
       $R_i^* = I_w \quad , \quad i = 1, \ldots, n$
2. Set $k = 0$, the iteration counter.
3. Set $x_b = [L^*, R^*], y_b = \text{fitness}(L^*, R^*, N_s)$. The function fitness() is presented in
   Algorithm 3 and it is used a measure of each interval.
4. Set $x_c = x_b, y_c = y_b$
5. For $i = 1, \ldots, N_{\text{eps}}$ do
   (a) Create a new random interval $x_t = (L_t, R_t)$ with the incorporation of of
       Algorithm 4 by calling $x_t = \text{newInterval}(L_c, R_c, p_c)$ .
   (b) Calculate $y_t = \text{fitness}(L_t, R_t, N_s)$ using Algorithm 3.
   (c) If $y_t < y_c$ then Set $x_c = x_t, y_c = y_t$
   (d) Else Set $x_c = x_t$ with probability $\min\left\{1, \exp\left(-\frac{f_t - f_c}{T_k}\right)\right\}$
   (e) End if
6. End For
7. Update the temperature using $T_{k+1} = T_k r_T$
8. Set $k = k + 1$
9. If $T_k \le \epsilon$ terminate and return $x_c = [L_c, R_c]$ as the discovered interval for the
   parameters of the neural network.
10. Go to step 5.
End function
```

Algorithm 3 The algorithm below is used to calculate the function value for a specified interval of parameters.

```
function fitness$(L, R, N_s)$
1. Produce the set $T = \{w_1, w_2, \ldots, w_{N_s}\}$ with $N_s$ random samples in $[L, R]$. Each sample
   is a set of parameters for the neural network and its created randomly in $[L, R]$.
2. Set $k = 0$
3. For $i = 1, \ldots, N_s$ do
   (a) Create a neural network $N\left(\overrightarrow{x}, \overrightarrow{w_i}\right)$.
   (b) Calculate the training error $f_i = \sum_{j=1}^{M}\left(N\left(\overrightarrow{x_j}, \overrightarrow{w_i}\right) - y_j\right)^2$ for the corresponding
       train set.
   (c) Obtain the quantity $b_i = B\left(N\left(\overrightarrow{x}, \overrightarrow{w}\right), a\right)$ using algorithm 1.
   (d) Set $k = k + f_i \times (1 + \lambda b_i)$,where $\lambda > 1$.
4. End For
5. Set $k = \frac{k}{N_s}$
6. Return $k$.
End function
```

Algorithm 4 The following procedure is used to produce new random intervals for the bounds of the parameters of the neural network.

```
Function newInterval$(L, R, p_c)$
1.  For $i = 1, \dots, n$ do
    (a) Set $\delta_1$ is a random number that could be 1 or -1.
    (b) Set $\delta_2$ a random number that could be 1 or -1.
    (c) Set $L_i^x = L_i + \delta_1 p_c r_1 L_i$, where $r_1 \in [0, 1]$ a random value.
    (d) Set $R_i^x = R_i + \delta_2 p_c r_2 R_i$, where $r_2 \in [0, 1]$ a random value.
2.  End For
3.  Return $x = [L^x, R^x]$ as the constructed interval.
End function
```

2.2. *The final training method*

After finding a promising range of values for the parameters of the artificial neural network, the optimization of these parameters within this range is performed. This optimization minimizes the error function and any optimization method could be used. In this work, the Genetic Algorithm method was chosen to be used. The Genetic Algorithm method was preferred over other local optimization techniques, such as the Adam method, because it is a global optimization method and aims to find the global minimum of an objective function with a greater guarantee for finding the optimal set of values for the weights of the artificial neural network. Furthermore, it has been used effectively in the past for training artificial neural networks. Of course, any other global optimization method could be used in place of Genetic Algorithms. The main steps of this genetic algorithm are provided subsequently:

1. **Initialization step**.
   (a) **Set** the parameters of the algorithm:
      i. N_c the number of uses chromosomes.
      ii. N_g the number of allowed generations.
      iii. p_s the selection rate, with $p_s \leq 1$
      iv. p_m the mutation rate, with $p_m \leq 1$
   (b) **Initialize** the chromosomes $g_i,\ i = 1, \dots, N_c$ as vectors of randomly selected values. Each chromosome has $n = (d + 2)H$ elements. The initialization is performed inside the bounds $\left[\overrightarrow{L^*}, \overrightarrow{R^*}\right]$, produced during the first phase of the method as described in subsection 2.1.
   (c) **Set** $k = 0$, as the generation counter.
2. **Fitness calculation step**.
   (a) **For** $i = 1, \dots, N_c$ **do**
      i. **Create** the artificial neural network $N_i\left(\overrightarrow{x}, \overrightarrow{g_i}\right)$ for the corresponding chromosome $\overrightarrow{g_i}$.
      ii. **Set** $f_i = \sum_{j=1}^{M}\left(N\left(\overrightarrow{x_j}, \overrightarrow{g_i}\right) - y_j\right)^2$ as the associated fitness value.
   (b) **End For**
3. **Application of the genetic operators** .
   (a) **Selection**: The top $(1 - p_s) \times N_c$chromosomes of the current population, as determined by their fitness values, are directly preserved and transferred to the next generation. The remaining chromosomes are replaced by newly generated individuals resulting from the application of crossover and mutation operators.
   (b) **Crossover:** During this procedure a series of $p_s \times N_s$ new chromosomes will be produced from the old ones. For each couple $(\widetilde{z}, \widetilde{w})$ of new chromosomes,

two parents (z, w) are selected from the current population with tournament selection. The new couple will be created using the following operations:

$$\begin{aligned} \tilde{z_i} &= r_i z_i + (1-r_i) w_i,\ i = 1,\dots,n \\ \tilde{w_i} &= r_i w_i + (1-r_i) z_i,\ i = 1,\dots,n \end{aligned} \tag{6}$$

The numbers r_i are randomly selected numbers in $[-0.5, 1.5]$ [64].

(c) **Mutation:** This procedure is applied in every element of each chromosome, where a random number $r \in [0,1]$ is selected. If $r \le p_m$ then the corresponding element g_{ij} of chromosome g_i is changed using the following equation:

$$g_{ij} = \begin{cases} g_{ij} + \Delta\left(\mathrm{t}, b_{g,i} - g_{ij}\right) & t = 0 \\ g_{ij} - \Delta\left(\mathrm{t}, g_{ij} - a_{g,i}\right) & t = 1 \end{cases} \tag{7}$$

Here t is a random number that can be 0 or 1 and the function $\Delta(\mathrm{t}, y)$ is defined as:

$$\Delta(t, y) = y\left(1 - \omega^{\left(1 - \frac{t}{N_t}\right)z}\right) \tag{8}$$

The value ω is a random number in $[0,1]$ and z is user defined parameter.

4. **Termination check**.
   (a) **Set** $k = k + 1$
   (b) **If** $k < N_g$ then go to Fitness Calculation step.
5. **Testing step**.
   (a) **Denote** as g^* the chromosome with the lowest fitness value and create the corresponding neural network $N\left(\overrightarrow{x}, \overrightarrow{g^*}\right)$.
   (b) **Train** the neural network $N\left(\overrightarrow{x}, \overrightarrow{g^*}\right)$ using a local search procedure. In the current work the BFGS variant of Powell [63] was selected.
   (c) **Apply** the neural network on the corresponding test set of the objective problem and report the test error.

3. Results

The validation of the proposed approach was conducted using a collection of benchmark datasets for classification and regression tasks, which are publicly available online through the following sources:

1. The UCI database, https://archive.ics.uci.edu/(accessed on 31 December 2025)[65]
2. The Keel website, https://sci2s.ugr.es/keel/datasets.php(accessed on 31 December 2025)[66].
3. The Statlib database, https://lib.stat.cmu.edu/datasets/index(accessed on 9 December 2025).

3.1. Experimental datasets

The classification datasets employed in the experimental evaluation are listed below:

1. **Appendictis** dataset, as provided in [67].
2. **Alcohol** dataset, which is a dataset regarding experiments on alcohol consumption [68].
3. **Australian** dataset, derived from bank transactions [69].
4. **Balance** dataset [70], which was used in a series of psychological experiments.
5. **Cleveland**, which was studied in a series of papers [71,72].
6. **Circular** dataset, which is an artificial dataset with two classes.

7. **Dermatology**, that provides measurements from dermatology problems [73].
8. **Ecoli**, which is related to protein problems [74].
9. **Glass** dataset, related to glass component analysis.
10. **Haberman**, a medical dataset that is used for to detect the presence of breast cancer.
11. **Hayes-roth** dataset [75].
12. **Heart**, used to predict the presence of heart diseases [76].
13. **HeartAttack**, a medical dataset used in heart diseases
14. **Housevotes**, a dataset related to the Congressional voting in USA [77].
15. **Ionosphere**, related to measurements from the ionosphere and studied in a series of papers [78,79].
16. **Liverdisorder**, a medical dataset [80,81].
17. **Lymography** dataset[82].
18. **(OK)Mammographic**, which is related to breast cancer detection [83].
19. **Parkinsons**, which is related to the detection of Parkinson's disease [84,85].
20. **Pima**, related to the detection of diabetes[86].
21. **Phoneme**, a dataset that contains sound measurements.
22. **Popfailures**, which is related to measurements regarding climate [87].
23. **Regions2**, a dataset used for the detection of liver problems [88].
24. **Saheart**, which is a medical dataset concerning heart diseases[89].
25. **Segment** dataset [90].
26. **Statheart**, a dataset related to the detection of heart diseases.
27. **Spiral**, an artificial dataset with two classes.
28. **Student**, which is a dataset regarding experiments in schools [91].
29. **Transfusion** dataset [92].
30. **Wdbc**, used to detect the breast cancer [93,94].
31. **Wine**, a dataset used to detect the quality of wines [95,96].
32. **EEG**, that contains EEG measurements [97,98]. From this dataset the following cases were studied: Z_F_S, ZO_NF_S and ZONF_S.
33. **Zoo**, a dataset used to classify animals in some categories [99] .

In addition, the following regression datasets were utilized in the experimental evaluation:

1. **Abalone**, regarding the detection of the age of abalones [100].
2. **Airfoil**, a dataset derived from NASA [101].
3. **Auto**, a dataset related to the fuel consumption in cars.
4. **BK**, related to the prediction of points scored by basketball players.
5. **BL**, used in various electricity experiments.
6. **Baseball**, related to the prediction of income of baseball players.
7. **Concrete**, a dataset which is related to civil engineering [102].
8. **DEE**, a dataset used to predict the prices of electricity.
9. **Friedman** dataset[103].
10. **FY,** related to fruit flies.
11. **HO**, a dataset derived from the STATLIB repository.
12. **Housing**, used to predict the prices of houses [104].
13. **Laser**, used in various experiments in physics.
14. **LW**, a dataset related to the weight of babes.
15. **Mortgage**, a dataset related to economics.
16. **PL** dataset, derived from the STALIB repository.
17. **Plastic**, a dataset used to detect problems in plastics.
18. **Quake**, which contains measurements from earthquakes.
19. **SN**, a benchmark dataset commonly employed in trellising and pruning studies.
20. **Stock**, regarding the prices of stocks.

21. **Treasury**, a dataset related to economics.

3.2. Experimental results

The experimental software was implemented in C++ using the freely available Optimus framework [105] Each experimental configuration was executed 30 independent times, with a distinct random seed assigned to each run. To ensure a reliable evaluation of the results, the ten-fold cross-validation methodology was employed. All experiments were repeated 30 times, and performance was quantified using the mean classification error for the classification datasets and the mean regression error for the regression datasets. The classification error is calculated using the following equation:

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

Here the set $T = \{x_i, y_i\},\ i = 1, \ldots, K$ represents the associated test set of the objective problem. Similarly, the regression error for the test set is given from the following equation:

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

The values for every parameter of the proposed algorithm are given in Table 1.

Table 1. The values for every experimental parameter.

PARAMETER	MEANING	VALUE
H	Number of weights	10
λ	Scale parameter	10.0
I_w	Weight parameter	10.0
T_0	Initial temperature	3.0
r_t	Decrease factor for temperature	0.95
Neps	Number of random interval produced	100
a	Scale factor	10.0
p_c	Perturbation factor	0.01
N_s	Number of random samples	25
N_c	Number of chromosomes	500
N_g	Number of allowed generations	500
p_s	Selection rate	0.90
p_m	Mutation rate	0.05
z	Mutation parameter	1.0

Moreover, the following notation is used in the tables that provide the experimental results:

1. The column DATASET provides the name of the dataset.
2. The column ADAM denotes the experimental results by the usage of the ADAM optimization method [22] to train a neural network having $H = 10$ processing nodes.
3. The column BFGS denotes the usage of the BFGS method to train an artificial neural network with $H = 10$ processing nodes.
4. The column GENETIC denotes the usage of Genetic Algorithm to train a neural network with $H = 10$ processing nodes.
5. The column RBF denotes the application of a Radial Basis Function (RBF) network [106,107] with $H = 10$ hidden nodes.

6. The column NEAT denotes the incorporation of the NEAT method (NeuroEvolution of Augmenting Topologies) [108].
7. The column PRUNE denotes the application of the OBS pruning method [109].
8. The column PROPOSED stands for the application of the current method.
9. The row AVERAGE stands for the the average classification or regression error.

The experimental results by the application of the previously mentioned machine learning methods to the classification datasets are depicted in Table 2. Also, the corresponding results for the regression datasets are shown in Table 3.

Table 2 reports classification error rates (lower is better) across 34 datasets for seven learning/training approaches, with the last row providing the average error per method. At the aggregate level, the proposed approach (PROPOSED) is clearly the best performer, achieving an average error of 20.57%, while the second-best average is obtained by GENETIC at 26.55%. This corresponds to an absolute improvement of 5.97 percentage points, i.e., an approximately 22.5% relative error reduction compared to GENETIC. The advantage remains consistent against the remaining baselines: relative to PRUNE (27.44%) the improvement is 6.87 points (~25.0% relative reduction), relative to RBF (29.42%) it is 8.85 points (~30.1%), relative to NEAT (32.11%) it is 11.54 points (~35.9%), and relative to ADAM/BFGS (34.48%/34.34%) it is about 13.9 points (~40% relative reduction). In addition, when considering variability across heterogeneous datasets, PROPOSED exhibits the lowest dispersion of errors (standard deviation $\approx$ 14.31), which is consistent with more stable behavior across different classification tasks.

From a per-dataset perspective (minimum error per row), PROPOSED attains the best result on 23 out of 34 datasets ($\approx$67.6%), indicating that its superiority is systematic rather than driven by a small subset of cases. Moreover, PROPOSED ranks within the top two methods on 31/34 datasets and within the top three on 33/34 datasets, highlighting strong rank consistency. In the datasets where PROPOSED is not the best, it is often very close to the winner (e.g., SPIRAL: 44.90% vs 44.87%, ALCOHOL: 15.99% vs 15.75%, POPFAILURES: 5.06% vs 4.79%), with a limited number of more pronounced gaps such as ECOLI (49.67% vs 43.44%) and DERMATOLOGY (14.83% vs 9.02%). Conversely, there are datasets where PROPOSED achieves large margins over the second-best method, notably SEGMENT (38.85% vs 49.75%), HEARTATTACK (18.97% vs 29.00%), HEART (18.37% vs 27.21%), and WINE (8.12% vs 16.62%). Overall, Table 2 supports that the proposed method delivers the best mean performance, the best average ranking, and the highest number of dataset-level wins, with only sporadic cases where alternative methods outperform it.

Table 2. Experimental results on the classification datasets using the series of the machine learning methods. Numbers in cells represent average classification error as measured on the corresponding test set.

DATASET	ADAM	BFGS	GENETIC	RBF	NEAT	PRUNE	PROPOSED
APPENDICITIS	16.50%	18.00%	24.40%	12.23%	17.20%	15.97%	15.50%
ALCOHOL	57.78%	41.50%	39.57%	49.32%	66.80%	15.75%	15.99%
AUSTRALIAN	35.65%	38.13%	32.21%	34.89%	31.98%	43.66%	27.22%
BALANCE	12.27%	8.64%	8.97%	33.53%	23.14%	9.00%	8.60%
CLEVELAND	67.55%	77.55%	51.60%	67.10%	53.44%	51.48%	44.48%
CIRCULAR	19.95%	6.08%	5.99%	5.98%	35.18%	12.76%	5.88%
DERMATOLOGY	26.14%	52.92%	30.58%	62.34%	32.43%	9.02%	14.83%
ECOLI	64.43%	69.52%	54.67%	59.48%	43.44%	60.32%	49.67%
GLASS	61.38%	54.67%	52.86%	50.46%	55.71%	66.19%	52.57%
HABERMAN	29.00%	29.34%	28.66%	25.10%	24.04%	29.38%	26.87%
HAYES-ROTH	59.70%	37.33%	56.18%	64.36%	50.15%	45.44%	34.23%
HEART	38.53%	39.44%	28.34%	31.20%	39.27%	27.21%	18.37%
HEARTATTACK	45.55%	46.67%	29.03%	29.00%	32.34%	29.26%	18.97%
HOUSEVOTES	7.48%	7.13%	6.62%	6.13%	10.89%	5.81%	4.96%
IONOSPHERE	16.64%	15.29%	15.14%	16.22%	19.67%	11.32%	10.17%
LIVERDISORDER	41.53%	42.59%	31.11%	30.84%	30.67%	49.72%	33.71%
LYMOGRAPHY	39.79%	35.43%	28.42%	25.50%	33.70%	22.02%	19.93%
MAMMOGRAPHIC	46.25%	17.24%	19.88%	21.38%	22.85%	38.10%	17.13%
PARKINSONS	24.06%	27.58%	18.05%	17.41%	18.56%	22.12%	14.58%
PIMA	34.85%	35.59%	32.19%	25.78%	34.51%	35.08%	27.90%
POPFAILURES	5.18%	5.24%	5.94%	7.04%	7.05%	4.79%	5.06%
REGIONS2	29.85%	36.28%	29.39%	38.29%	33.23%	34.26%	31.48%
SAHEART	34.04%	37.48%	34.86%	32.19%	34.51%	37.70%	32.15%
SEGMENT	49.75%	68.97%	57.72%	59.68%	66.72%	60.40%	38.85%
SPIRAL	47.67%	47.99%	48.66%	44.87%	48.66%	50.38%	44.90%
STATHEART	44.04%	39.65%	27.25%	31.36%	44.36%	28.37%	21.07%
STUDENT	5.13%	7.14%	5.61%	5.49%	10.20%	10.84%	4.50%
TRANSFUSION	25.68%	25.84%	24.87%	26.41%	24.87%	29.35%	23.59%
WDBC	35.35%	29.91%	8.56%	7.27%	12.88%	15.48%	4.21%
WINE	29.40%	59.71%	19.20%	31.41%	25.43%	16.62%	8.12%
Z_F_S	47.81%	39.37%	10.73%	13.16%	38.41%	17.91%	7.70%
ZO_NF_S	47.43%	43.04%	21.54%	9.02%	43.75%	15.57%	6.66%
ZONF_S	11.99%	15.62%	4.36%	4.03%	5.44%	3.27%	2.78%
ZOO	14.13%	10.70%	9.50%	21.93%	20.27%	8.53%	6.90%
AVERAGE	**34.48%**	**34.34%**	**26.55%**	**29.42%**	**32.11%**	**27.44%**	**20.57%**

The significance levels shown in Figure 2 were obtained via R-based analyses on the classification experiment tables, aiming to verify that the observed performance differences are statistically reliable rather than due to random variation. The overall Friedman test yields p=2.89×10^{-11}, which is extremely small and therefore strongly rejects the null hypothesis that all methods perform equivalently. This indicates that, across the set of datasets, genuine performance differences exist among the considered models and motivates post-hoc pairwise comparisons against the proposed approach.

The pairwise results confirm that the proposed method differs significantly from each baseline. In particular, the comparisons ADAM vs PROPOSED and BFGS vs PROPOSED produce p=1.78×10^{-8} and p=5.17×10^{-9}, respectively, providing very strong evidence of a difference (well beyond the $p<0.0001$ threshold). The PRUNE vs PROPOSED comparison is also highly decisive (p=2.07×10^{-5}, i.e., $p<0.0001$). For GENETIC and RBF, the p-values are larger but remain below 0.01 (p=0.0076 and p=0.0085), which corresponds to a "highly significant" difference. Overall, Figure 2 supports that the proposed model's superiority is not only reflected in the raw error rates, but is also substantiated by strong statistical significance against all competing baselines.

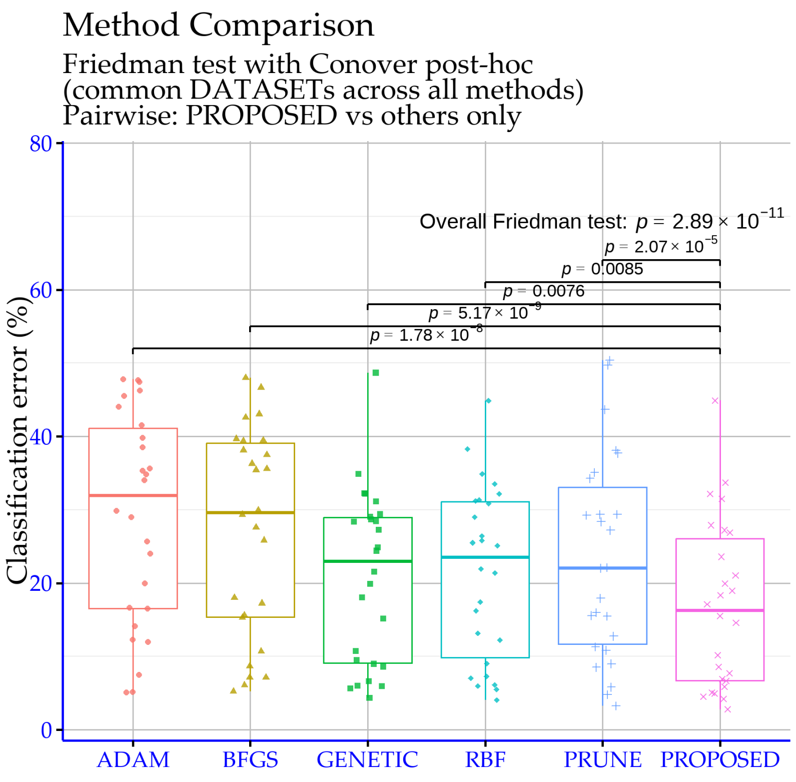

Figure 2. Statistical significance levels (Friedman and post-hoc) for classification experiments across learning models

Also, a comparison between the genetic algorithm and the proposed one for different number of processing nodes is outlined in Figure 3, that clearly demonstrates the potential of the current work. The numbers in graph indicate average classification error for all classification datasets that participate in the experiments.

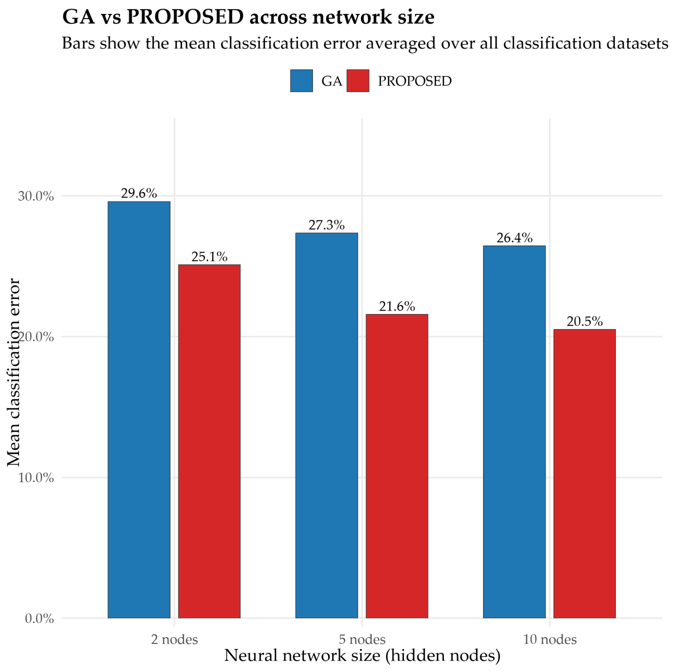

Figure 3. Genetic algorithm versus the proposed method across network size.

In Figure 4, the computed parameter intervals of the neural network are reported as left and right bounds (L and R) obtained by the Simulated Annealing bound-optimization stage, for three datasets with different dimensionality (WINE: 150 parameters, WDBC: 320, DERMATOLOGY: 360). The visualization highlights that the bounds do not change uniformly; instead, they vary substantially across parameters and across datasets, indicating that the proposed stage does not enforce a fixed range but identifies problem-dependent search regions. Most intervals cross zero and remain centered close to 0, while their widths can become very large and exhibit occasional extreme expansions, particularly for DERMATOLOGY where a subset of parameters attains markedly wider intervals. As shown in Figure [bounds_sa.png], this variability supports the claim that the resulting bounds adapt to each dataset's structure and scale, thereby providing a tailored search space for the subsequent training phase.

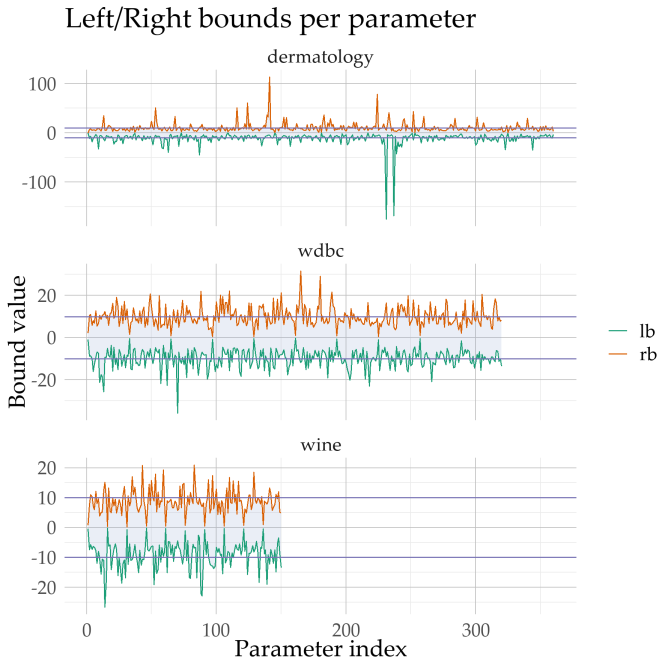

Figure 4. The estimated intervals for the neural network parameters from the proposed Simulated Annealing for three datasets.

In the regression part of Table 3, errors are reported in absolute units and span a very wide dynamic range (from approximately 10^{-3} up to hundreds), meaning that a few large-error cases can dominate average performance. Under this regime, the proposed approach (PROPOSED) delivers the most favorable aggregate outcome, with the lowest mean error of 5.33, compared to 9.31 for GENETIC and 10.02 for RBF. The reduction from 9.31 to 5.33 corresponds to a 3.98-unit absolute gain, i.e., about a 42.8% relative improvement over the best competing average (GENETIC). Importantly, PROPOSED also exhibits the strongest robustness against extreme values, showing the smallest variability across datasets (standard deviation ≈ 13.75). This point matters in regression benchmarking because heavy-tailed errors can materially affect the mean and may reveal instability. In contrast, ADAM and BFGS show very large worst-case outcomes (e.g., 180.89 and 302.43 on STOCK), which inflates their averages (22.46 and 30.29, respectively) and indicates sensitivity to high-scale or difficult regression settings.

Looking at dataset-level outcomes (row-wise minima), PROPOSED attains the best result or ties for best on 14 out of 21 regression datasets, including 11 outright wins and 3 top ties. Beyond wins, its rank consistency is strong: PROPOSED falls within the top two methods on 18/21 datasets and within the top three on 20/21 datasets, suggesting that its advantage is not driven by a small number of isolated successes. Several datasets show substantive margins rather than marginal differences, including HOUSING (26.76 vs 43.26 for the runner-up), TREASURY (0.20 vs 2.02), MORTGAGE (0.12 vs 1.45), and BASEBALL (58.86 vs 77.90). The few cases where PROPOSED is not the best do not overturn the overall pattern: on ABALONE it is essentially tied with the best value (4.31 vs 4.30), BL is dominated by RBF (0.013), and FY is the only dataset where PROPOSED falls outside the top three, yet the absolute gap remains small (0.057 vs 0.038). Overall, Table 2 indicates that the proposed method achieves the strongest balance of low mean error, frequent top performance, and reduced exposure to severe outliers, which is particularly relevant for regression evaluation across heterogeneous datasets.

Table 3. Experimental results on the regression datasets using the list of the provided machine learning methods. Numbers in cells represent average regression error on each test set.

DATASET	ADAM	BFGS	GENETIC	RBF	NEAT	PRUNE	PROPOSED
ABALONE	4.30	5.69	7.17	7.37	9.88	7.88	4.31
AIRFOIL	0.005	0.003	0.003	0.27	0.067	0.002	0.002
AUTO	70.84	60.97	12.18	17.87	56.06	75.59	12.07
BK	0.0252	0.28	0.027	0.02	0.15	0.027	0.025
BL	0.622	2.55	5.74	0.013	0.05	0.027	0.032
BASEBALL	77.90	119.63	103.60	93.02	100.39	94.50	58.86
CONCRETE	0.078	0.066	0.0099	0.011	0.081	0.0077	0.004
DEE	0.63	2.36	1.013	0.17	1.512	1.08	0.23
FRIEDMAN	22.90	1.263	1.249	7.23	19.35	8.69	2.58
FY	0.038	0.19	0.65	0.041	0.08	0.042	0.057
HO	0.035	0.62	2.78	0.03	0.169	0.03	0.01
HOUSING	80.99	97.38	43.26	57.68	56.49	52.25	26.76
LASER	0.03	0.015	0.59	0.03	0.084	0.007	0.003
LW	0.028	2.98	1.90	0.03	0.03	0.02	0.016
MORTGAGE	9.24	8.23	2.41	1.45	14.11	12.96	0.12
PL	0.117	0.29	0.29	2.118	0.09	0.032	0.022
PLASTIC	11.71	20.32	2.791	8.62	20.77	17.33	2.21
QUAKE	0.07	0.42	0.04	0.07	0.298	0.04	0.04
SN	0.026	0.40	2.95	0.027	0.174	0.032	0.026
STOCK	180.89	302.43	3.88	12.23	12.23	39.08	4.30
TREASURY	11.16	9.91	2.93	2.02	15.52	13.76	0.20
AVERAGE	**22.46**	**30.29**	**9.31**	**10.02**	**14.65**	**15.40**	**5.33**

Figure 5 reports statistical significance levels for the regression experiments, based on p-values computed through R scripts. The overall Friedman test yields p=1.55×10^{-4}, which is well below 0.001, indicating that the compared methods do not behave equivalently across the regression datasets and that genuine performance differences exist. This justifies examining post-hoc pairwise comparisons against the proposed method.

The pairwise outcomes reveal a more mixed pattern than in the classification case, which is expected in regression settings where absolute-error scales can vary substantially across datasets. The strongest evidence of a difference in favor of the proposed approach is observed against BFGS ($p = 3.35 \times 10^{-4}$), which falls in the extremely significant range ($p < 0.001$). The comparison with GENETIC yields $p = 0.0113$, i.e., significant at the 0.05 level but not highly significant, implying a reliable yet weaker separation. In contrast, ADAM, RBF, and PRUNE produce p-values of 0.0585, 0.1462, and 0.0898, respectively, all above 0.05, meaning that differences versus the proposed method are not statistically supported under the standard threshold. Overall, Figure 5 suggests that, for regression, the proposed method exhibits statistically substantiated advantages primarily over BFGS and, to a lesser extent, over GENETIC, while differences relative to the remaining baselines are not strong enough to rule out random experimental variability.

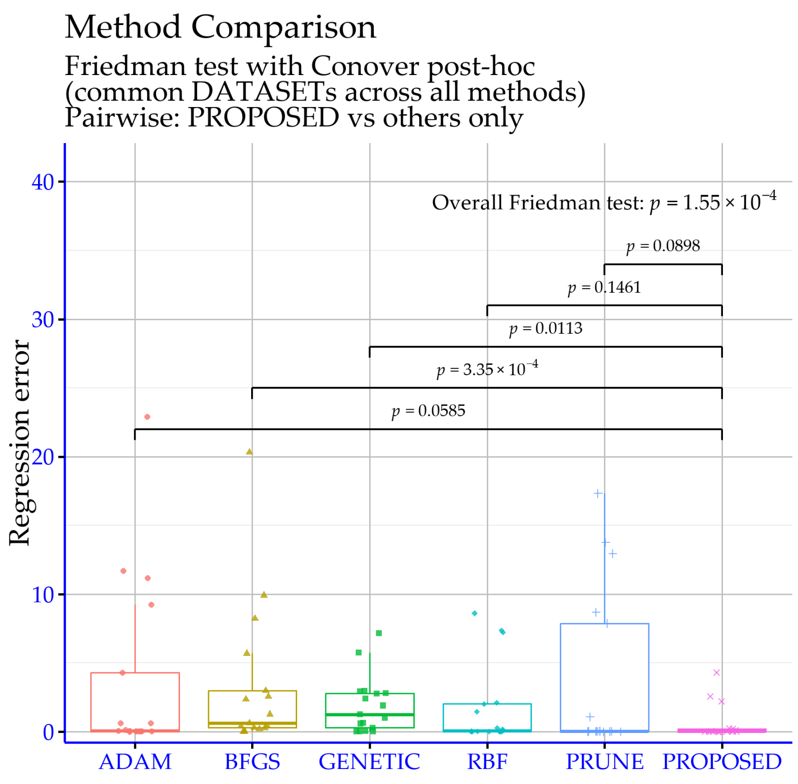

Figure 5. Statistical significance levels (Friedman and post-hoc) for regression experiments across learning models

Also, in Figure 6 an indicative plot is presented for the comparison of the training process between the genetic algorithm and the proposed method for the regression dataset MORTGAGE.

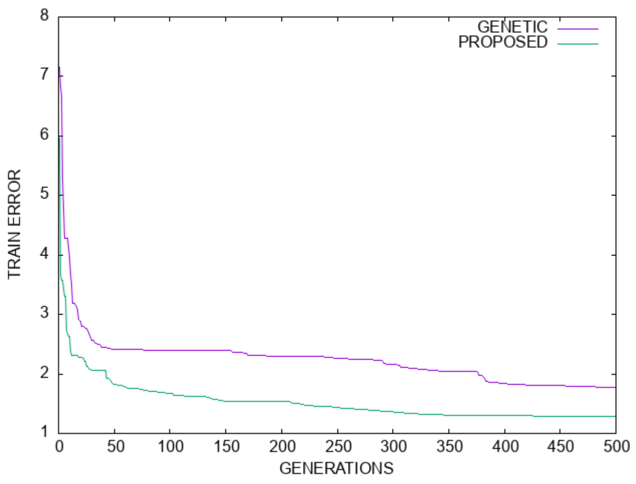

Figure 6. An indicative comparison of the training process between the genetic algorithm and the proposed method for the MORTGAGE dataset.

The Figure 7 reports mean runtime as a function of the neural network's number of nodes, comparing the Genetic Algorithm against the proposed method. Both runtimes increase with network size, but the proposed method consistently exhibits a much higher computational cost. Specifically, for 2 nodes the Genetic Algorithm requires 11.34 on average versus 213.21 for the proposed method; for 5 nodes the corresponding values are 29.18 versus 420.43, and for 10 nodes they are 57.95 versus 790.35. The gap remains large in all cases, and the relative overhead decreases only slightly as network size grows, without changing the overall conclusion. Overall, the results document a clear trade-off: the proposed method is substantially more time- consuming, and its use is therefore justified when improvements in accuracy or generalization outweigh the additional computational burden.

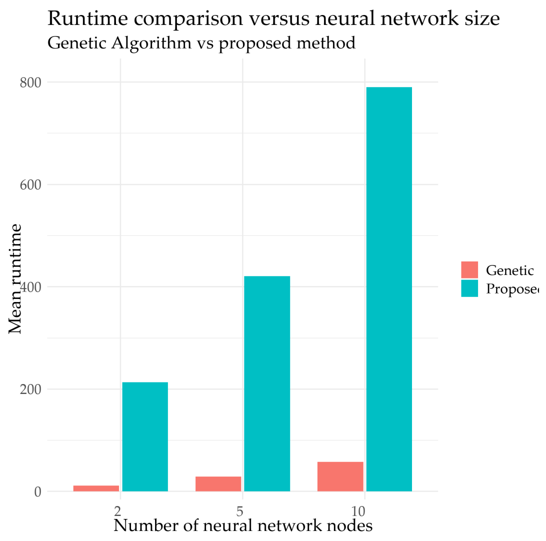

Figure 7. Average execution time comparison between the genetic algorithm and the proposed method for different number of weights for the neural network.

3.3. *Experiments with the perturbation parameter p_c*

An additional experiment was performed to determine the stability of the proposed technique to changes in the perturbation factor p_c. The experimental results for this experiment on the classification datasets are depicted in Table 4 and for regression datasets in Table 5.

Table 4 provides a targeted sensitivity study of the proposed classifier with respect to the crossover-related parameter p_c, reporting error rates (lower is better) on 34 classification datasets for three settings (p_c=0.01, 0.02, 0.05). The averages at the bottom of the table are extremely close, yet they consistently favor p_c=0.05: 20.33% versus 20.54% for p_c=0.02 and 20.57% forp_c=0.01. The absolute gain relative to $p_c = 0.01$ is only 0.24 percentage points (approximately a 1.2% relative reduction), which suggests that the method's performance is not highly sensitive to moderate changes in p_c. This limited sensitivity is also reflected by the central tendency and spread: the medians are nearly identical (about 17.75%, 17.85%, and 17.90%), and the dispersion across datasets remains comparable for all three settings (standard deviation around 14%), indicating that p_c mainly shifts performance on particular datasets rather than reshaping the overall distribution.

A more informative view emerges from the dataset-wise minima. The setting p_c=0.05 achieves the lowest error on 13 out of 34 datasets and ties for best on one additional dataset (STUDENT), i.e., 14/34 top outcomes. In comparison, p_c=0.01 is best on 11 datasets, while p_c=0.02 is best on 9 datasets plus one top tie. Hence, p_c=0.05 is the strongest default choice in terms of both average performance and frequency of best results, but the table also highlights clear dataset-specific preferences. For instance, p_c=0.05 yields notable improvements on LIVERDISORDER (33.71% to 30.26%), APPENDICITIS (15.50% to 12.40%), DERMATOLOGY (14.83% to 11.92%), AUSTRALIAN (27.22% to 24.58%), and ECOLI (49.67% to 47.15%). Conversely, p_c=0.01 is distinctly advantageous on SEGMENT (38.85% vs 44.21-44.72), SPIRAL (44.90% vs 45.29-47.77), and WINE (8.12% vs 10.65-11.23). Overall, Table 4 indicates that p_c acts as a fine-grained control parameter: p_c=0.05 optimizes mean behavior and the count of best-case outcomes, while smaller values such as p_c=0.01 can be preferable for specific datasets.

Table 4. Experimental results using the proposed method and a variety values for the perturbation factor p_c.

DATASET	$p_c = 0.01$	$p_c = 0.02$	$p_c = 0.05$
APPENDICITIS	15.50%	13.70%	12.40%
ALCOHOL	15.99%	16.10%	15.49%
AUSTRALIAN	27.22%	27.55%	24.58%
BALANCE	8.60%	8.02%	8.76%
CLEVELAND	44.48%	45.24%	46.04%
CIRCULAR	5.88%	6.94%	6.91%
DERMATOLOGY	14.83%	12.97%	11.92%
ECOLI	49.67%	47.55%	47.15%
GLASS	52.57%	50.34%	50.81%
HABERMAN	26.87%	27.27%	27.33%
HAYES-ROTH	34.23%	37.69%	35.00%
HEART	18.37%	18.00%	18.33%
HEARTATTACK	18.97%	20.10%	19.67%
HOUSEVOTES	4.96%	4.48%	3.39%
IONOSPHERE	10.17%	7.72%	9.06%
LIVERDISORDER	33.71%	32.74%	30.26%
LYMOGRAPHY	19.93%	21.14%	22.29%
MAMMOGRAPHIC	17.13%	17.70%	17.47%
PARKINSONS	14.58%	13.63%	13.84%
PIMA	27.90%	27.20%	26.42%
POPFAILURES	5.06%	4.82%	5.71%
REGIONS2	31.48%	30.86%	29.81%
SAHEART	32.15%	31.91%	32.23%
SEGMENT	38.85%	44.21%	44.72%
SPIRAL	44.90%	45.29%	47.77%
STATHEART	21.07%	19.08%	20.70%
STUDENT	4.50%	3.95%	3.95%
TRANSFUSION	23.59%	23.74%	24.07%
WDBC	4.21%	4.38%	4.18%
WINE	8.12%	11.23%	10.65%
Z_F_S	7.70%	8.27%	6.57%
ZO_NF_S	6.66%	6.28%	5.54%
ZONF_S	2.78%	2.60%	2.82%
ZOO	6.90%	5.80%	5.40%
AVERAGE	**20.57%**	**20.54%**	**20.33%**

Figure 8 indicates that varying the pc parameter does not produce statistically detectable differences in the proposed model's performance on the classification datasets. The overall Friedman test yields p=0.6556, hence the null hypothesis of equivalent settings is not rejected. Moreover, the pairwise comparisons p_c=0.01 vs p_c=0.05 and p_c=0.02 vs p_c=0.05 both give p=1, confirming no evidence of separation. Therefore, within the tested range, p_c behaves as a practically neutral tuning choice, and the selection can be guided by secondary considerations (e.g., stability or computational cost). In summary, the three pc settings are statistically indistinguishable for classification under the reported tests.

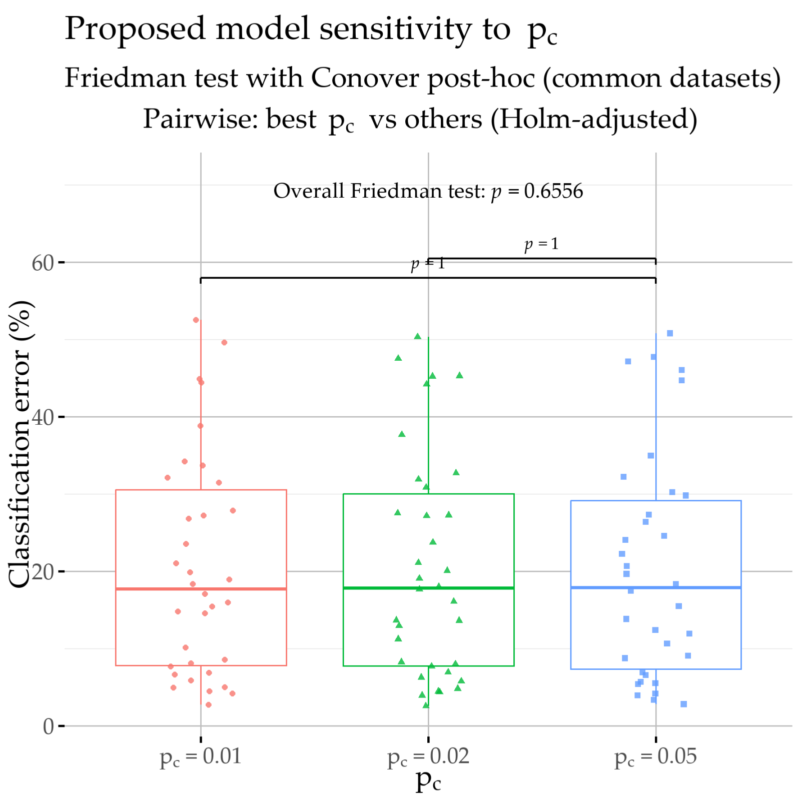

Figure 8. Statistical comparison of p_c settings for the proposed model on classification datasets

Table 5 evaluates the proposed method on regression tasks under three settings of the parameter p_c(0.01, 0.02, 0.05), reporting absolute errors per dataset. A key characteristic of these results is the pronounced scale heterogeneity: several datasets yield very small errors (on the order of 10^{-3} to 10^{-1}), whereas a small number of datasets produce much larger values (most notably BASEBALL and HOUSING). Consequently, the mean is strongly influenced by a few large-error cases, while the median better reflects the "typical" dataset behavior. Under the mean criterion, p_c=0.02 provides the best overall outcome (average error ≈ 5.27), followed closely by p_c=0.01 (≈ 5.33), where asp_c=0.05 is higher (≈ 5.40). This ordering is consistent with the medians as well (0.050 forp_c=0.02, 0.057 for p_c=0.01, 0.060 for p_c=0.05), indicating that the 0.02 setting tends to reduce central errors slightly, without causing a major shift in the overall distribution.

At the dataset level, p_c=0.02 emerges as the most consistently favorable choice. It achieves the lowest error with a strict advantage on 11 datasets and, when ties are included, it matches the best value on 17 out of 21 datasets. The clearest gains for p_c=0.02 occur on datasets that also affect the mean, such as AUTO (11.55 vs 12.07/11.87), BL (0.006 vs 0.032/0.010), BASEBALL (57.83 vs 58.86/60.52), MORTGAGE (0.079 vs 0.12/0.085), QUAKE (0.007 vs 0.04/0.011), and STOCK (3.93 vs 4.30/4.51). In contrast, p_c=0.05provides a clear advantage only on a limited subset, primarily BK (0.019), HOUSING (26.53), and TREASURY (0.17), while p_c=0.01 is strictly best only on PLASTIC (2.21). Additionally, several datasets are effectively insensitive to p_c (AIRFOIL, CONCRETE, HO, PL), where all settings yield identical outcomes. Overall, for regression performance as summarized in Table 5, p_c=0.02 is the most reliable default setting in terms of both average error and frequency of best results, with p_c=0.05 being preferable in specific datasets andp_c=0.01 offering isolated advantages.

Table 5. Experimental results on the regression datasets using the proposed method and a series of values for the perturbation factor p_c.

DATASET	$p_c = 0.01$	$p_c = 0.02$	$p_c = 0.05$
ABALONE	4.31	4.31	4.34
AIRFOIL	0.002	0.002	0.002
AUTO	12.07	11.55	11.87
BK	0.025	0.024	0.019
BL	0.032	0.006	0.01
BASEBALL	58.86	57.83	60.52
CONCRETE	0.004	0.004	0.004
DEE	0.23	0.22	0.23
FRIEDMAN	2.58	2.47	2.79
FY	0.057	0.05	0.06
HO	0.01	0.01	0.01
HOUSING	26.76	27.66	26.53
LASER	0.003	0.003	0.007
LW	0.016	0.015	0.019
MORTGAGE	0.12	0.079	0.085
PL	0.022	0.022	0.022
PLASTIC	2.21	2.30	2.24
QUAKE	0.04	0.007	0.011
SN	0.026	0.024	0.026
STOCK	4.30	3.93	4.51
TREASURY	0.20	0.18	0.17
AVERAGE	**5.33**	**5.27**	**5.40**

Figure 9 shows no statistically significant differences among pc settings on the regression datasets. The Friedman test yields p=0.092 (>0.05), so equivalence is not rejected. The pairwise comparisons likewise provide no evidence of separation. Therefore, the tested p_c values can be treated as practically equivalent for regression. 512 513 514 515

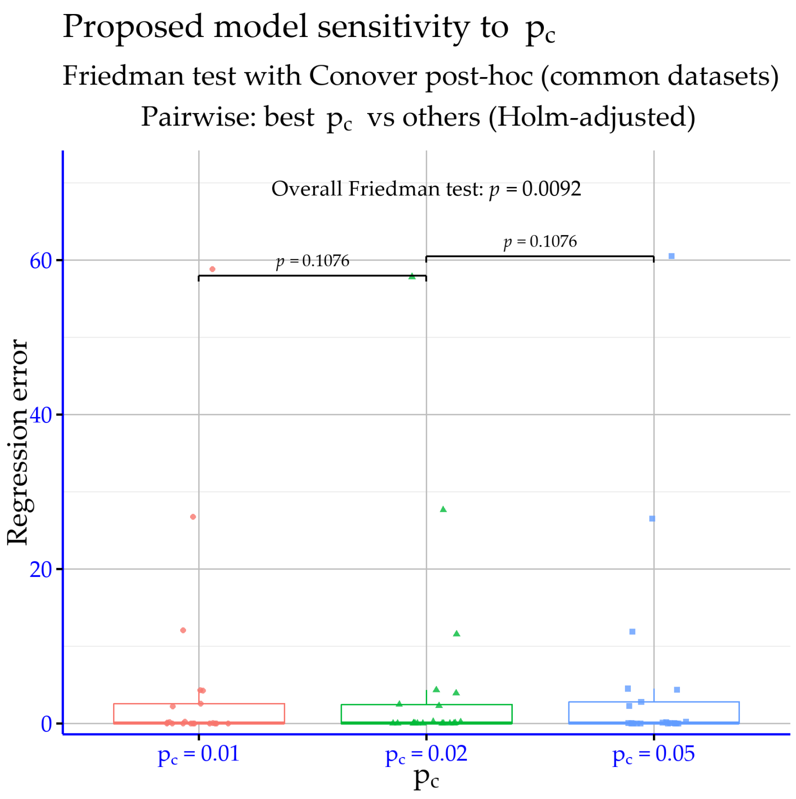

Figure 9. Statistical comparison of p_c settings for the proposed model on regression datasets

3.4. Experiments with the weight parameter I_w

In order to determine the stability of the proposed method under different initialization conditions, another experiment was conducted using it in which various values for the initialization parameter I_w were tested. The experimental results for the classification datasets are shown in Table 6 and for regression datasets in Table 7.

Table 6 examines the sensitivity of the proposed classifier to the parameter I_w using three settings (1, 10, 20) across 34 classification datasets, where lower error rates indicate better performance. The aggregate trend is monotonic and unfavorable as I_w increases: the average error rises from 19.45% at I_w=1 to 20.57% at I_w=10 and 21.15% at I_w=20. In practical terms, I_w=1 yields an absolute advantage of 1.12 percentage points over I_w=10 and 1.70 points over I_w=20, corresponding to approximately 5.5% and 8.0% relative error reductions, respectively. The same pattern is reflected in the distribution's center, with medians of about 17.17% (I_w=1), 17.75% (I_w=10), and 18.49% (I_w=20). Meanwhile, the spread across datasets remains comparable (standard deviation close to 14% for all three settings), suggesting that the degradation with higher I_w is not driven by a small number of extreme cases but by a consistent upward shift in errors.

A dataset-wise view further supports this conclusion. The setting I_w=1 achieves the lowest error on 22 of the 34 datasets, whereas I_w=10 and I_w=20 each win on 6 datasets, with no ties for best. Importantly, larger $I_{(w)}$values are not uniformly detrimental: there are clear instances where I_w=20 improves performance, such as CLEVELAND (43.20%), CIRCULAR (5.14%), PIMA (26.46%), and SPIRAL (43.86%). However, these gains are offset by pronounced losses on other datasets, including DERMATOLOGY where I_w=1 is markedly superior (6.03% vs 13.97-14.83), SEGMENT (33.09% vs 38.85-47.02), and WINE (6.82% vs 8.12-12.00). Overall, Table 6 indicates that I_w=1 is the most reliable default for classification, while higher settings behave more like specialized adjustments that can benefit particular datasets but tend to reduce average performance across the benchmark suite.

Table 6. Experimental results on the classification dataset using the proposed method and different values for the weight parameter I_w.

DATASET	$I_w = 1$	$I_w = 10$	$I_w = 20$
APPENDICITIS	13.30%	15.50%	15.40%
ALCOHOL	13.19%	15.99%	18.54%
AUSTRALIAN	28.41%	27.22%	29.33%
BALANCE	9.66%	8.60%	9.16%
CLEVELAND	47.48%	44.48%	43.20%
CIRCULAR	6.65%	5.88%	5.14%
DERMATOLOGY	6.03%	14.83%	13.97%
ECOLI	45.06%	49.67%	50.09%
GLASS	49.86%	52.57%	52.57%
HABERMAN	26.83%	26.87%	27.47%
HAYES-ROTH	33.62%	34.23%	39.39%
HEART	16.93%	18.37%	18.48%
HEARTATTACK	18.53%	18.97%	21.00%
HOUSEVOTES	3.26%	4.96%	5.96%
IONOSPHERE	8.17%	10.17%	12.17%
LIVERDISORDER	30.09%	33.71%	32.44%
LYMOGRAPHY	17.43%	19.93%	18.50%
MAMMOGRAPHIC	17.41%	17.13%	17.19%
PARKINSONS	16.16%	14.58%	13.84%
PIMA	31.12%	27.90%	26.46%
POPFAILURES	5.09%	5.06%	4.87%
REGIONS2	29.84%	31.48%	31.16%
SAHEART	30.37%	32.15%	32.69%
SEGMENT	33.09%	38.85%	47.02%
SPIRAL	46.67%	44.90%	43.86%
STATHEART	17.63%	21.07%	18.82%
STUDENT	3.60%	4.50%	4.88%
TRANSFUSION	23.92%	23.59%	23.88%
WDBC	4.98%	4.21%	4.52%
WINE	6.82%	8.12%	12.00%
Z_F_S	8.03%	7.70%	7.80%
ZO_NF_S	5.60%	6.66%	6.70%
ZONF_S	2.72%	2.78%	2.76%
ZOO	3.60%	6.90%	7.70%
AVERAGE	**19.45%**	**20.57%**	**21.15%**

Figure 10 suggests that the Iw parameter has a marginal yet detectable effect on the classification datasets, as the overall Friedman test yields p=0.0423 and slightly rejects the null of equivalent settings at the 0.05 level. However, the pairwise comparisons do not support strong separation: I_w=1 vs I_w=10 gives p=0.2834 (not significant) andI_w=1 vs I_w=20 gives p=0.069 (also not significant, but close to 0.05). This indicates that differences are modest and distributed across settings rather than producing a clear, strongly significant pairwise contrast. Practically, Iw behaves as a secondary tuning parameter with limited impact, where an overall difference is detectable but not pronounced in simple post-hoc tests. Hence, selecting I_w can be guided by mean performance or stability, acknowledging that the statistical effects are weak.

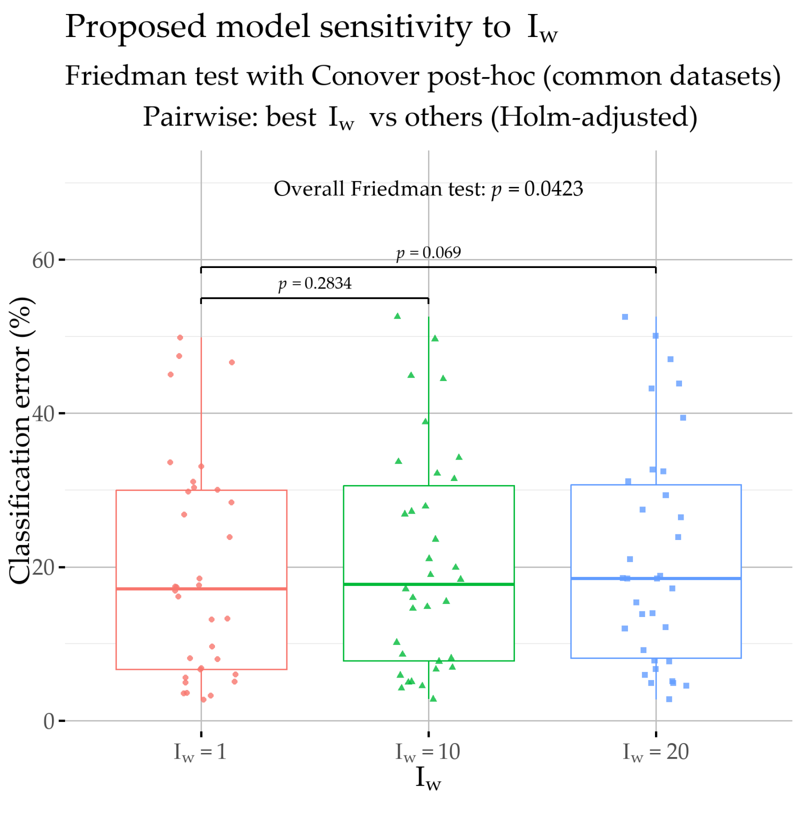

Figure 10. Statistical comparison of I_w settings for the proposed model on classification datasets

Table 7 reports a regression sensitivity analysis of the proposed method with respect to I_w(1, 10, 20) using absolute error values. At the aggregate level, differences are modest: the average error is 6.10 for I_w=1, 5.33 forI_w=10, and 5.46 for I_w=20. Thus, I_w=10 is the best overall setting, improving the mean by 0.77 relative to I_w=1(≈12.6% reduction) and by 0.13 relative to I_w=20(≈2.4%). Given the heterogeneous error scales across datasets, these mean differences are influenced by a small number of high-magnitude cases, so it is also informative to consider the per-dataset pattern.

At the dataset level, no single setting dominates uniformly, rather, I_w behaves as a tuning parameter. I_w=10 attains the minimum error on 8 out of 21 datasets and is close to the best on several others, while I_w=20 is best on 7 datasets and I_w=1 on 6 datasets. The most visible gains favoring I_w=10 occur on AUTO (12.07 vs 16.88), BASEBALL (58.86 vs 67.47), and TREASURY (0.20 vs 1.98), whereas I_w=20 is clearly preferable on ABALONE (4.22), AIRFOIL (0.001), FRIEDMAN (2.07), MORTGAGE (0.088), and STOCK (4.13). Conversely, I_w=1 is best on BK (0.021), BL (0.003), FY (0.041), HO (0.008), HOUSING (23.12), LASER (0.005), and LW (0.012). Overall, Table 7 suggests I_w=10 as a reasonable default for regression under the mean criterion, while the optimal choice can vary by dataset without producing large shifts in the overall performance picture.

Table 7. Experimental results with the application of the proposed method to the regression datasets, using a variety of values for the weight parameter I_w .

DATASET	$i_w = 1$	$i_w = 10$	$i_w = 20$
ABALONE	4.52	4.31	4.22
AIRFOIL	0.003	0.002	0.001
AUTO	16.88	12.07	11.10
BK	0.021	0.025	0.028
BL	0.003	0.032	0.02
BASEBALL	67.47	58.86	60.41
CONCRETE	0.005	0.004	0.004
DEE	0.23	0.23	0.23
FRIEDMAN	3.28	2.58	2.07
FY	0.041	0.057	0.12
HO	0.008	0.01	0.01
HOUSING	23.12	26.76	29.55
LASER	0.005	0.003	0.003
LW	0.012	0.016	0.018
MORTGAGE	1.08	0.12	0.088
PL	0.034	0.022	0.023
PLASTIC	3.94	2.21	2.23
QUAKE	0.04	0.04	0.04
SN	0.025	0.026	0.026
STOCK	5.37	4.30	4.13
TREASURY	1.98	0.20	0.30
AVERAGE	**6.10**	**5.33**	**5.46**

Figure 11 indicates that varying the $I_w$parameter does not produce statistically significant differences on the regression datasets. The overall Friedman test reports $p = 0.3050$, therefore the null hypothesis of equivalent settings is not rejected under the standard 0.05 threshold. The pairwise comparisons are clearly non-significant as well, with $p = 0.9856$ for I_w=10 vs I_w=20 and $p = 0.7928$ for I_w=1 vs I_w=20. Such large p-values imply practically indistinguishable behavior among the examined settings, with no evidence of a consistent winner. Hence, for regression, I_w can be selected based on secondary considerations because its statistical impact appears negligible.

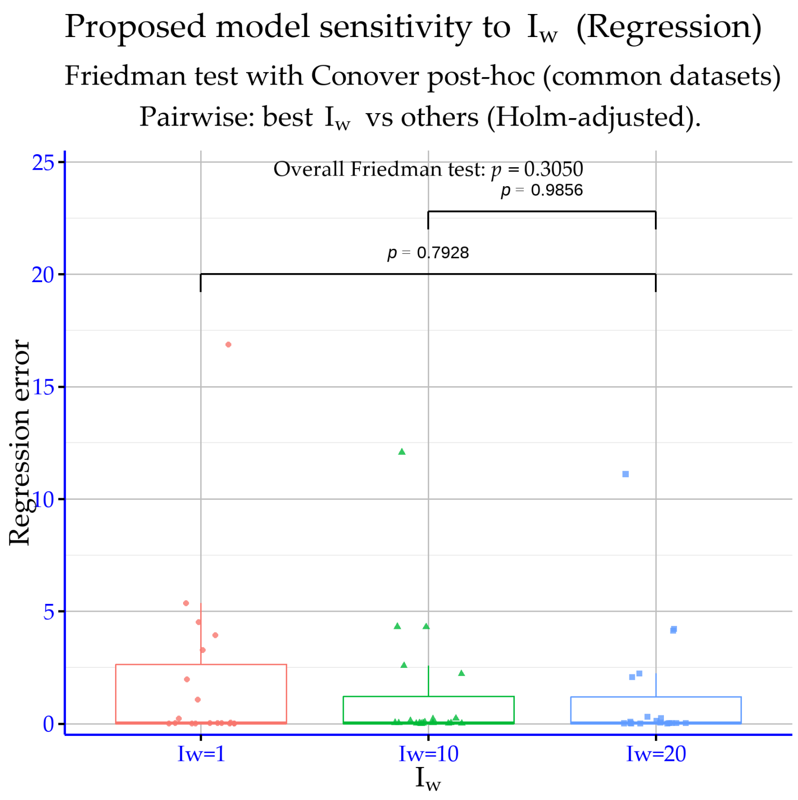

Figure 11. Statistical comparison of I_w settings for the proposed model on regression datasets

3.5. Experiments with initialization methods

Additionally, three method from the related literature were used as the initialization methods for the weights of the neural network:

1. The Xavier method [110], which sets the variance of the weights based on both the number of input and output connections in the hidden layer.
2. The HE method [111], which scales the weight variance according to the number of input connections
3. The LeCun method [112], that sets the weight variance inversely proportional to the number of input connections.

These weight initialization techniques were used by a genetic algorithm to train the artificial neural network and the results for the classification datasets are outlined in Table 8 and for the regression datasets in Table 9.

Table 8isolates the role of the initialization distribution within a genetic-training setting and compares it directly against the proposed machine-learning model on the same set of classification datasets, using error rate as the performance measure. In terms of averages, the four genetic variants differ in a noticeable yet bounded manner, ranging from 26.55% for the uniform initialization down to 24.69% for the LeCun initialization, a spread of about 1.86 percentage points. This indicates that initialization matters, but it does not, by itself, reshape the overall performance profile. The proposed machine-learning model attains a substantially lower mean error of 20.57%, which is about 4.12 percentage points below even the strongest genetic initialization (LeCun), corresponding to roughly a 16.7% relative reduction in mean error.

The dataset-level pattern confirms that this advantage is not merely an artifact of aggregation. The proposed machine-learning model achieves the lowest error on 26 out of 34 datasets, showing consistent superiority over all genetic initialization choices across most of the benchmark suite. The limited cases where a genetic initialization is better occur on BALANCE, CIRCULAR, DERMATOLOGY, GLASS, HABERMAN, LIVERDISORDER, MAMMOGRAPHIC, and REGIONS2. Even among these, some gaps are marginal, while others are more pronounced, suggesting that dataset structure can selectively favor specific initialization schemes. At the same time, on several challenging datasets the proposed machine-learning model yields clear reductions relative to every genetic variant, including APPENDICITIS, ALCOHOL, HEART, HEARTATTACK, SEGMENT, WDBC, WINE, and ZO_NF_S. Overall, Table 8 supports that initialization choice within the genetic framework has a measurable but secondary effect, whereas the proposed machine-learning model delivers a consistently stronger outcome both in average error and in the frequency of best per-dataset results.

Table 8. Experimental results on the classification datasets using the genetic algorithm as the training method and series of initialization methods.

DATASET	GENETIC UNIFORM	GENETIC XAVIER	GENETIC XE	GENETIC LECUN	PROPOSED
APPENDICITIS	24.40%	23.00%	23.10%	23.30%	15.50%
ALCOHOL	39.57%	24.04%	26.94%	23.81%	15.99%
AUSTRALIAN	32.21%	33.91%	34.39%	34.06%	27.22%
BALANCE	8.97%	8.53%	8.76%	8.56%	8.60%
CLEVELAND	51.60%	58.18%	60.42%	57.76%	44.48%
CIRCULAR	5.99%	4.76%	6.00%	5.13%	5.88%
DERMATOLOGY	30.58%	16.00%	16.43%	13.80%	14.83%
ECOLI	54.67%	54.33%	54.30%	51.39%	49.67%
GLASS	52.86%	52.67%	54.90%	51.86%	52.57%
HABERMAN	28.66%	27.80%	27.10%	26.27%	26.87%
HAYES-ROTH	56.18%	37.31%	47.23%	38.00%	34.23%
HEART	28.34%	29.52%	31.74%	30.22%	18.37%
HEARTATTACK	29.03%	32.37%	32.67%	31.50%	18.97%
HOUSEVOTES	6.62%	8.56%	9.26%	8.39%	4.96%
IONOSPHERE	15.14%	14.49%	14.69%	15.97%	10.17%
LIVERDISORDER	31.11%	33.18%	34.59%	33.24%	33.71%
LYMOGRAPHY	28.42%	28.36%	26.07%	25.14%	19.93%
MAMMOGRAPHIC	19.88%	16.82%	18.43%	17.24%	17.13%
PARKINSONS	18.05%	18.37%	18.26%	17.32%	14.58%
PIMA	32.19%	33.83%	34.80%	34.26%	27.90%
POPFAILURES	5.94%	8.04%	7.69%	7.56%	5.06%
REGIONS2	29.39%	31.73%	30.16%	30.53%	31.48%
SAHEART	34.86%	34.24%	34.02%	33.91%	32.15%
SEGMENT	57.72%	49.84%	47.11%	45.55%	38.85%
SPIRAL	48.66%	47.43%	49.18%	48.49%	44.90%
STATHEART	27.25%	29.96%	27.78%	27.04%	21.07%
STUDENT	5.61%	6.60%	6.05%	6.50%	4.50%
TRANSFUSION	24.87%	24.96%	24.81%	24.62%	23.59%
WDBC	8.56%	8.97%	9.11%	7.61%	4.21%
WINE	19.20%	22.12%	19.18%	17.35%	8.12%
Z_F_S	10.73%	17.33%	15.37%	15.63%	7.70%
ZO_NF_S	21.54%	14.50%	13.38%	13.88%	6.66%
ZONF_S	4.36%	3.70%	3.96%	3.90%	2.78%
ZOO	9.50%	11.30%	10.70%	9.70%	6.90%
AVERAGE	**26.55%**	**25.49%**	**25.84%**	**24.69%**	**20.57%**

Table 9 reports regression results in absolute error units and compares four genetic-training variants that differ only in the initialization distribution against the proposed machine-learning model. The data indicate that initialization within the genetic framework has a tangible impact on aggregate performance, as the genetic mean error changes from 9.31 with UNIFORM to 7.24 with XAVIER and 7.32 with LECUN. Nevertheless, the proposed machine-learning model achieves the lowest mean error, 5.33, which is 1.91 units below the best genetic mean, corresponding to an approximately 26% relative reduction. This suggests that the observed gains cannot be explained merely by a more favorable random initialization, but reflect a more effective overall training mechanism.

At the dataset level, the proposed machine-learning model is dominant: it delivers the lowest error on 17 out of 21 datasets and ties for best on one additional dataset, while being outperformed on only three datasets. The advantage is particularly pronounced on problems where the genetic variants exhibit large errors or strong sensitivity to initialization, such as BL, HO, LW, SN, MORTGAGE, and HOUSING, where the proposed approach yields substantially smaller values than all genetic initializations. The few exceptions occur on AIRFOIL and STOCK, and also on FRIEDMAN where the genetic variants attain lower error. Overall, Table 9 supports that while initialization choice improves genetic training to some extent, it does not match the consistently stronger regression performance of the proposed machine-learning model across the benchmark suite.

Table 9. Experimental results on the regression datasets using the genetic algorithm as the training method of the neural network and a series of initialization techniques.

DATASET	GENETIC UNIFORM	GENETIC XAVIER	GENETIC XE	GENETIC LECUN	PROPOSED
ABALONE	7.17	4.47	4.59	4.43	4.31
AIRFOIL	0.003	0.001	0.001	0.001	0.002
AUTO	12.18	13.46	14.63	12.99	12.07
BK	0.027	0.16	0.05	0.06	0.025
BL	5.74	2.13	1.82	2.75	0.032
BASEBALL	103.60	79.39	82.76	79.29	58.86
CONCRETE	0.0099	0.026	0.015	0.015	0.004
DEE	1.013	0.59	0.74	0.84	0.23
FRIEDMAN	1.249	1.23	1.29	1.23	2.58
FY	0.65	0.29	0.92	0.52	0.057
HO	2.78	0.69	0.89	0.52	0.01
HOUSING	43.26	41.76	46.21	42.81	26.76
LASER	0.59	0.004	0.004	0.004	0.003
LW	1.90	0.63	0.37	0.43	0.016
MORTGAGE	2.41	0.97	1.18	0.83	0.12
PL	0.29	0.034	0.39	0.09	0.022
PLASTIC	2.791	2.90	3.36	3.22	2.21
QUAKE	0.04	0.09	0.24	0.38	0.04
SN	2.95	0.49	1.43	0.62	0.026
STOCK	3.88	1.85	2.04	1.87	4.30
TREASURY	2.93	0.92	1.15	0.88	0.20
AVERAGE	**9.31**	**7.24**	**7.81**	**7.32**	**5.33**

Figure 12 reports statistical significance levels for the classification experiments comparing the proposed machine-learning model against four genetic-training variants that differ by their initialization distribution. The overall Friedman test yields $p = 2.48 \times 10^{-9}$, an extremely small value indicating that performance differences among the methods are real and cannot be attributed to random variability. In other words, across the dataset suite, the alternative genetic initializations and the proposed approach lead to measurably different performance behavior.

The pairwise comparisons against the proposed model show statistically supported differences in all cases. Specifically, GENETIC (UNIFORM) vs the proposed model and GENETIC (XAVIER) vs the proposed model both yield $p = 1.25 \times 10^{-5}$, providing very strong evidence of a difference well below the $p < 0.0001$ threshold. The GENETIC (XE) comparison is even more decisive with $p = 6.79 \times 10^{-8}$, also far below 0.0001. The GENETIC (LECUN) comparison yields $p = 0.0149$, which remains significant at the 0.05 level but is clearly weaker, consistent with this variant being the most competitive among the genetic baselines. Overall, Figure 12 confirms that the proposed method's advantage over the genetic variants is not only reflected in mean error rates, but is also statistically substantiated across the full set of classification experiments.

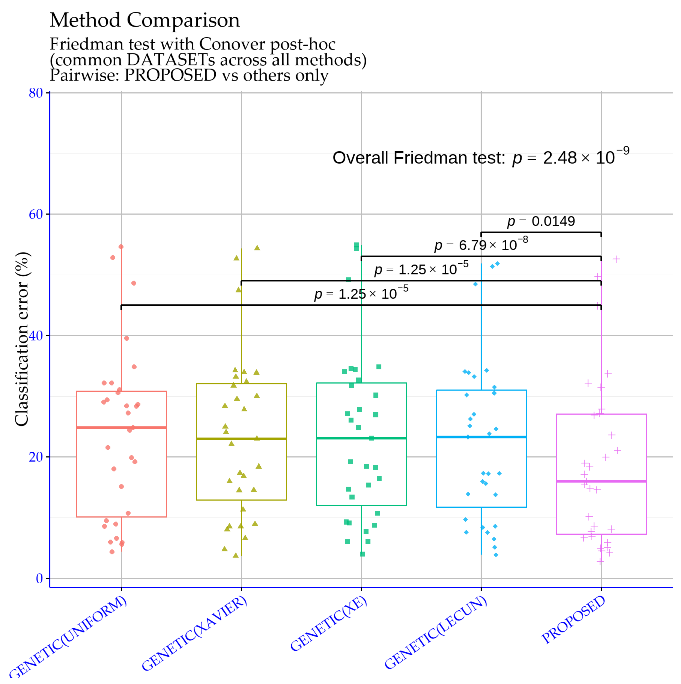

Figure 12. Statistical comparison on the results performed on the classification datasets using different initialization methods.

Figure 13 reports R-based statistical significance levels for comparisons between the proposed machine-learning model and four genetic variants that differ by their initialization distribution, using the regression experiment results. The overall Friedman test yields $p = 1.06 \times 10^{-5}$, which strongly rejects the hypothesis that all methods behave equivalently across the regression datasets. Therefore, at the suite level, the genetic-initialization choices and the proposed approach lead to performance differences that are unlikely to be explained by random variation.

The pairwise comparisons show that the strength of separation depends on the specific genetic variant. GENETIC(UNIFORM) vs the proposed model and GENETIC(XE) vs the proposed model yield $p = 1.03 \times 10^{-4}$ and $p = 4.98 \times 10^{-4}$, respectively, both below 0.001 and thus extremely significant, providing strong evidence that the proposed method differs from these two initializations. In contrast, the comparisons against GENETIC(XAVIER) and GENETIC(LECUN) give $p = 0.0941$, which is above 0.05, so a statistically significant difference is not supported under the standard threshold. Overall, Figure 13 suggests that the proposed approach is statistically superior relative to the weaker genetic initializations, while against the more competitive genetic choices the observed differences are not large enough to rule out random experimental variability.

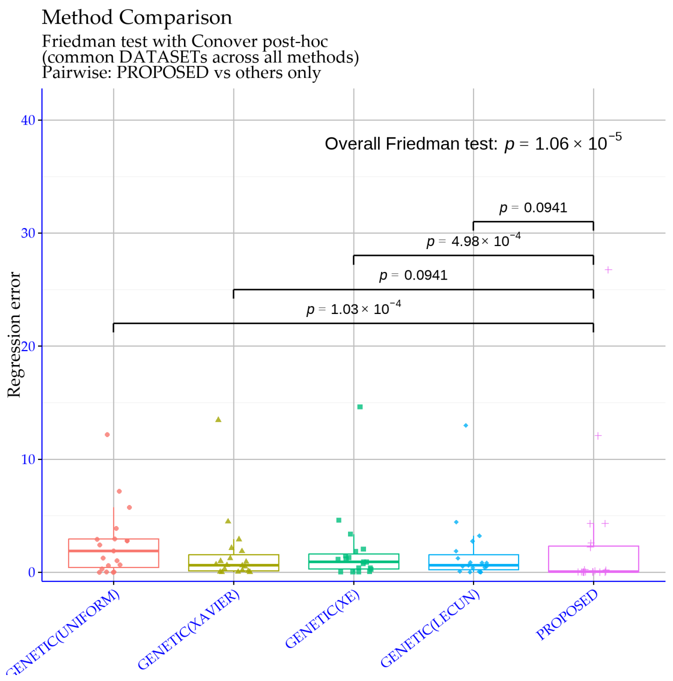

Figure 13. Statistical comparison for the experiments performed on the regression datasets using different initialization methods.

3.6. Experiments with the cooling strategy

An additional experiment was executed where the cooling strategy was altered, in order to verify the robustness of the proposed method. In this experiment the following strategies were used:

1. Exponential decreasing (EXP), the temperature is decreased using the following equation:
$$T = T_0 a^k,$$
where k defines the current iteration.
2. Logarithmical multiplicative cooling (LOG), where the following formula is used:
$$T_k = \frac{T_0}{1 + a\log(1+k)}$$
3. Linear multiplicative cooling (LINEAR), using the following formula:
$$T_k = \frac{T_0}{1 + ak}$$
4. Quadratic multiplicative cooling (QUAD), with the following formula:
$$T_k = \frac{T_0}{1 + ak^2}$$

The experimental results for the classification datasets are reported in Table 10and for regression datasets in Table 11.

In 10, the classification error rate (%) is reported for 34 datasets under four cooling schedules (EXP, LOG, LINEAR, QUAD) within the proposed model, where lower values indicate better performance. In terms of average error, EXP provides the best overall result (20.57%), while LOG (20.88%) and LINEAR (20.87%) are essentially tied and very close, and QUAD yields the highest average (21.01%). Although the average differences are small, the best schedule can be dataset-dependent, and some datasets exhibit higher sensitivity to the cooling choice (e.g., SEGMENT and WINE). As shown in Figure 14, there is no statistical significance according to the p-values. 689 690

Table 10. Experimental results on the classification datasets using the proposed method and a series of cooling strategies.

DATASET	EXP	LOG	LINEAR	QUAD
APPENDICITIS	15.50%	14.50%	14.60%	15.60%
ALCOHOL	15.99%	13.25%	14.16%	15.01%
AUSTRALIAN	27.22%	27.55%	28.51%	29.60%
BALANCE	8.60%	8.15%	8.56%	8.50%
CLEVELAND	44.48%	46.07%	46.45%	44.45%
CIRCULAR	5.88%	6.64%	5.88%	4.06%
DERMATOLOGY	14.83%	13.69%	16.20%	16.89%
ECOLI	49.67%	49.61%	50.61%	50.39%
GLASS	52.57%	50.38%	51.05%	51.38%
HABERMAN	26.87%	27.23%	27.27%	27.10%
HAYES-ROTH	34.23%	37.85%	37.16%	38.00%
HEART	18.37%	18.78%	17.74%	18.37%
HEARTATTACK	18.97%	20.60%	20.30%	20.73%
HOUSEVOTES	4.96%	5.87%	5.35%	6.18%
IONOSPHERE	10.17%	9.77%	9.09%	11.66%
LIVERDISORDER	33.71%	32.38%	32.15%	33.03%
LYMOGRAPHY	19.93%	22.57%	21.71%	22.79%
MAMMOGRAPHIC	17.13%	17.62%	17.20%	17.17%
PARKINSONS	14.58%	14.32%	14.53%	14.74%
PIMA	27.90%	27.95%	27.49%	27.84%
POPFAILURES	5.06%	5.41%	5.04%	5.31%
REGIONS2	31.48%	30.89%	30.63%	30.42%
SAHEART	32.15%	32.37%	32.59%	32.59%
SEGMENT	38.85%	45.28%	42.98%	39.51%
SPIRAL	44.90%	45.37%	44.73%	43.92%
STATHEART	21.07%	18.67%	19.41%	19.93%
STUDENT	4.50%	4.35%	4.38%	4.53%
TRANSFUSION	23.59%	24.03%	23.24%	23.26%
WDBC	4.21%	4.20%	4.77%	4.41%
WINE	8.12%	10.18%	11.71%	11.73%
Z_F_S	7.70%	7.80%	8.30%	8.03%
ZO_NF_S	6.66%	7.46%	6.50%	6.98%
ZONF_S	2.78%	2.68%	2.64%	2.80%
ZOO	6.90%	6.30%	6.80%	7.20%
AVERAGE	**20.57%**	**20.88%**	**20.87%**	**21.01%**

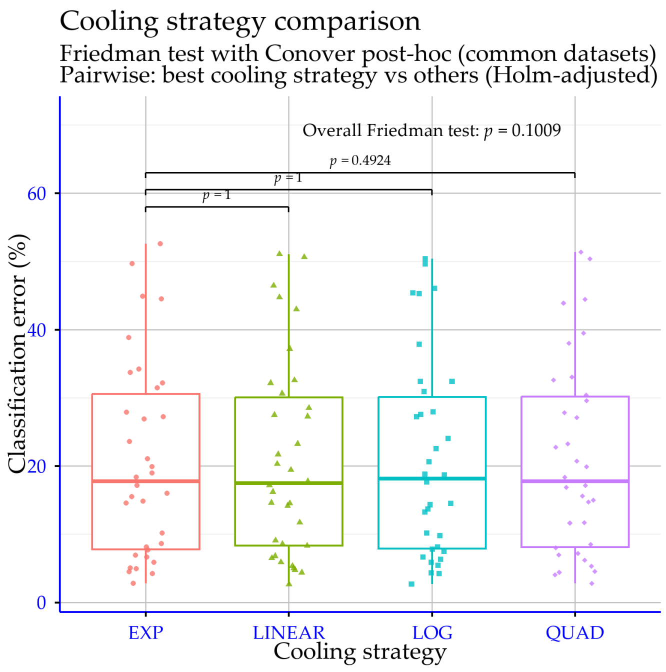

Figure 14. Statistical comparison for the obtained results on the classification datasets using the proposed method and a series of cooling strategies.

In Table 11, the reported values are the errors (lower is better) on 21 regression datasets, obtained by using four cooling schedules (EXP, LOG, LINEAR, QUAD) within the proposed model. If we focus on the overall picture, LOG has the lowest average error (5.16), EXP and LINEAR are essentially the same (5.33), and QUAD is slightly higher (5.42). In other words, the average differences are modest and do not suggest a single schedule that clearly dominates everywhere.

At the dataset level, however, the picture is less uniform. There are datasets where the cooling choice seems to matter more such as BL, HOUSING, FRIEDMAN, and STOCK because the gaps between schedules become noticeably larger. In contrast, datasets like CONCRETE, HO, and LASER remain almost unchanged across schedules, indicating that performance there is largely insensitive to how temperature is decreased. If a single, simple default must be used without per-dataset tuning, LOG is the most reasonable choice because it yields the best average. Figure 15 indicates that the p-values do not support statistically significant differences among the four cooling schedules.

Table 11. Experimental results on the regression datasets using the proposed method and a series of cooling strategies .

DATASET	EXP	LOG	LINEAR	QUAD
ABALONE	4.31	4.30	4.27	4.35
AIRFOIL	0.002	0.002	0.002	0.001
AUTO	12.07	11.79	11.43	12.33
BK	0.025	0.027	0.022	0.028
BL	0.032	0.015	0.008	0.03
BASEBALL	58.86	57.43	57.98	58.70
CONCRETE	0.004	0.004	0.004	0.004
DEE	0.23	0.22	0.23	0.22
FRIEDMAN	2.58	2.47	2.39	2.01
FY	0.057	0.049	0.058	0.15
HO	0.01	0.01	0.01	0.01
HOUSING	26.76	24.87	28.17	28.08
LASER	0.003	0.003	0.003	0.003
LW	0.016	0.015	0.015	0.03
MORTGAGE	0.12	0.09	0.09	0.04
PL	0.022	0.021	0.022	0.022
PLASTIC	2.21	2.35	2.29	2.26
QUAKE	0.04	0.04	0.05	0.04
SN	0.026	0.026	0.025	0.025
STOCK	4.30	4.45	4.57	5.29
TREASURY	0.20	0.26	0.20	0.24
AVERAGE	**5.33**	**5.16**	**5.33**	**5.42**

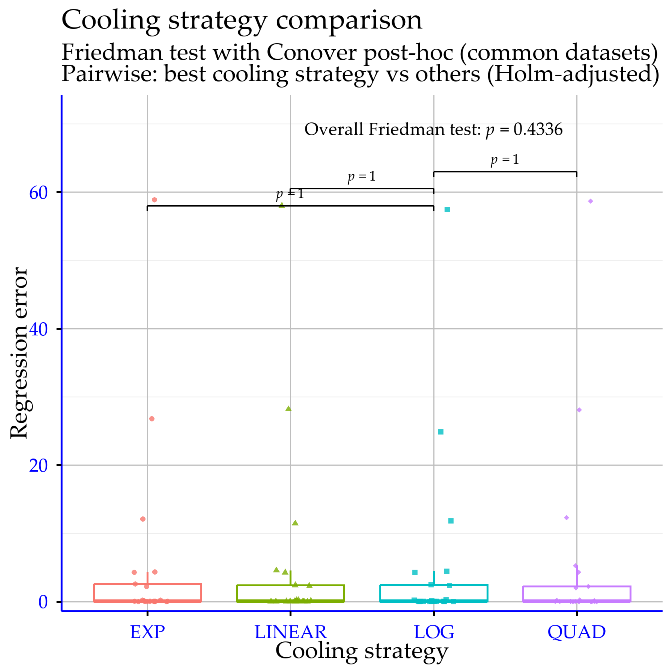

Figure 15. Statistical comparison for the obtained results on the regression datasets using the proposed method and a series of cooling strategies.

4. Conclusions

The experimental evidence indicates that the proposed method should be interpreted as a training pipeline rather than as a minor optimizer tweak. Its key design choice is a preliminary bound-selection stage that constrains the admissible weight ranges before the main optimization. This stage is implemented via a Simulated Annealing variant and is motivated by limiting regimes in which sigmoid units saturate, since large-magnitude inputs can yield near-constant outputs and degrade effective learning dynamics and generalization. Final training is then carried out inside the selected bounds through a Genetic optimization stage followed by local refinement with BFGS, so the intended benefit arises from combining a restricted search domain with a global stochastic search and a local convergence phase.

On classification benchmarks, the reported results are consistent with a systematic advantage. The proposed model achieves the best aggregate performance with an average classification error of 20.57%, while the closest competing average is GENETIC at 26.55%, corresponding to a 5.97 percentage-point absolute gain (approximately 22.5% relative error reduction). The advantage is also reported to remain sizable against the remaining baselines and is accompanied by the lowest dispersion of errors (standard deviation ≈ 14.31), which is compatible with more stable behavior across heterogeneous classification tasks. At the dataset level, PROPOSED attains the best result on 23 of 34 datasets and ranks within the top two on 31/34 datasets, supporting that the superiority is not driven by a small subset of cases.

The statistical analysis for classification further supports that these differences are unlikely to be attributable to random variation. The overall Friedman test yields $p = 2.89 \times 10^{-11}$, strongly rejecting the hypothesis of equivalent performance across methods. Post-hoc comparisons against PROPOSED are significant for all main baselines, with extremely small p-values against ADAM and BFGS, a highly decisive result against PRUNE, and still highly significant differences against GENETIC and RBF (p-values below 0.01).

For regression, the overall picture is positive but requires more conservative phrasing. PROPOSED attains the lowest mean absolute error (MAE = 5.33) and the overall Friedman test remains significant $\left(p = 1.55 \times 10^{-4}\right)$, indicating that genuine performance differences exist across the regression suite. However, pairwise evidence is more nuanced: the strongest statistically supported advantage is observed primarily against BFGS $\left(p = 3.35 \times 10^{-4}\right)$ and more weakly (but still significantly) against GENETIC ($p = 0.0113$), whereas differences relative to ADAM, RBF, and PRUNE are not supported at the 0.05 level. In addition, regression errors span a wide dynamic range and can be heavy-tailed, so robustness and variability across datasets materially affect the interpretation of mean outcomes.

Sensitivity experiments help identify which settings appear critical under the tested conditions. Variations of pc are reported not to yield measurable statistical differences (for both classification and regression), suggesting that pc is not a critical parameter within the explored range. In contrast, Iw shows a marginal overall effect in classification (overall $p = 0.0423$), but without strong pairwise confirmation, implying modest and distributed differences rather than a clear, sharply separated setting. This supports the interpretation that the dominant performance driver is the bound-selection mechanism and the overall pipeline structure, rather than fine tuning of auxiliary parameters.

A central and practically relevant trade-off is computational cost. The Simulated Annealing bound-selection phase requires repeated evaluations and may dominate runtime. This is directly reflected in the reported runtime comparison as a function of network size: for 2 nodes, GA averages 11.34 versus 213.21 for PROPOSED, for 5 nodes, 29.18 versus 420.43, and for 10 nodes, 57.95 versus 790.35. Consequently, the method is best positioned for settings where accuracy and robustness gains justify higher offline computation, rather than as a default choice under strict training-time constraints. Finally, the current validation targets sigmoid feed-forward networks and a saturation-driven rationale, it has not yet been demonstrated that the same gains transfer to deep networks or truly high-dimensional parameterizations without additional evaluation and potentially redesigning the underlying bound-selection criterion.

Several directions follow naturally for future work. Firstly, the proposed technique could be applied to other cases of artificial neural networks such as, for example, deep neural networks, although the required execution time would be quite long due to the complexity and number of parameters in these machine learning models. One possible solution to this problem could be the incorporation of parallel versions of the Simulated Annealing algorithm [113,114] that can take advantage of the modern computational environments. Second, the fitness definition used during the Simulated Annealing phase should be redesigned to incorporate explicit generalization criteria, such as validation error or robustness to noise, so that the bound-selection stage more directly targets out-of-sample performance. Third, the cost-benefit trade-off should be quantified systematically, and CPU or GPU parallelization should be explored, since the Simulated Annealing phase requires repeated evaluations and may dominate the overall runtime.

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 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).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

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

References

1. Müller, B., Reinhardt, J., & Strickland, M. T. (2012). Neural networks: an introduction. Springer Science & Business Media.
2. Gurney, K. (2018). An introduction to neural networks. CRC press.
3. I.G. Tsoulos, D. Gavrilis, E. Glavas, Neural network construction and training using grammatical evolution, Neurocomputing **72**, pp. 269-277, 2008.
4. S. Guarnieri, F. Piazza, A. Uncini, Multilayer feedforward networks with adaptive spline activation function, IEEE Transactions on Neural Networks **10**, pp. 672-683, 1999.
5. Ö.F. Ertuğrul, A novel type of activation function in artificial neural networks: Trained activation function, Neural Networks **99**, pp. 148-157, 2018.
6. A. D. Rasamoelina, F. Adjailia, P. Sinčák, A Review of Activation Function for Artificial Neural Network, In: 2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI), Herlany, Slovakia, pp. 281-286, 2020.
7. Pandelea, A. E., Budescu, M., & Covatariu, G. (2015). Image processing using artificial neural networks. Buletinul Institutului Politehnic din lasi. Sectia Constructii, Arhitectura, 61(4), 9.
8. Tealab, A. (2018). Time series forecasting using artificial neural networks methodologies: A systematic review. Future Computing and Informatics Journal, 3(2), 334-340.
9. 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.
10. P. Baldi, K. Cranmer, T. Faucett et al, Parameterized neural networks for high-energy physics, Eur. Phys. J. C **76**, 2016.
11. 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.
12. M.B. Kia, S. Pirasteh, B. Pradhan B. et al, An artificial neural network model for flood simulation using GIS: Johor River Basin, Malaysia, Environ Earth Sci **67**, pp. 251–264, 2012.
13. A.K. Yadav, S.S. Chandel, Solar radiation prediction using Artificial Neural Network techniques: A review, Renewable and Sustainable Energy Reviews **33**, pp. 772-781, 2014.
14. M.A. Getahun, S.M. Shitote, C. Zachary, Artificial neural network based modelling approach for strength prediction of concrete incorporating agricultural and construction wastes, Construction and Building Materials **190**, pp. 517-525, 2018.
15. M. Chen, U. Challita, W. Saad, C. Yin and M. Debbah, Artificial Neural Networks-Based Machine Learning for Wireless Networks: A Tutorial, IEEE Communications Surveys & Tutorials **21**, pp. 3039-3071, 2019.

16. 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.
17. Ronadl Bartzatt, Prediction of Novel Anti-Ebola Virus Compounds Utilizing Artificial Neural Network (ANN), Chemistry Faculty Publications **49**, pp. 16-34, 2018.
18. K. Peta, J. Żurek, Prediction of air leakage in heat exchangers for automotive applications using artificial neural networks, In: 2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York, NY, USA, pp. 721-725, 2018.
19. K. Vora, S. Yagnik, A survey on backpropagation algorithms for feedforward neural networks, International Journal of Engineering Development and Research **1**, pp. 193-197, 2014.
20. Pajchrowski, T., Zawirski, K., & Nowopolski, K. (2014). Neural speed controller trained online by means of modified RPROP algorithm. IEEE transactions on industrial informatics, 11(2), 560-568.
21. Hermanto, R. P. S., & Nugroho, A. (2018). Waiting-time estimation in bank customer queues using RPROP neural networks. Procedia Computer Science, 135, 35-42.
22. 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.
23. Reynolds, J., Rezgui, Y., Kwan, A., & Piriou, S. (2018). A zone-level, building energy optimisation combining an artificial neural network, a genetic algorithm, and model predictive control. Energy, 151, 729-739.
24. Sharma, D. K., Hota, H. S., Brown, K., & Handa, R. (2022). Integration of genetic algorithm with artificial neural network for stock market forecasting. International Journal of System Assurance Engineering and Management, 13(Suppl 2), 828-841.
25. Das, G., Pattnaik, P. K., & Padhy, S. K. (2014). Artificial neural network trained by particle swarm optimization for non-linear channel equalization. Expert Systems with Applications, 41(7), 3491-3496.
26. Sexton, R. S., Dorsey, R. E., & Johnson, J. D. (1999). Beyond backpropagation: using simulated annealing for training neural networks. Journal of Organizational and End User Computing (JOEUC), 11(3), 3-10.
27. 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.
28. Karaboga, D., & Akay, B. (2007, June). Artificial bee colony (ABC) algorithm on training artificial neural networks. In 2007 IEEE 15th Signal Processing and Communications Applications (pp. 1-4). IEEE.
29. Mosavi, M. R., Khishe, M., & Ghamgosar, A. (2016). Classification of sonar data set using neural network trained by gray wolf optimization. Neural Network World, 26(4), 393.
30. R.S. Sexton, B. Alidaee, R.E. Dorsey, J.D. Johnson, Global optimization for artificial neural networks: A tabu search application. European Journal of Operational Research **106**, pp. 570-584, 1998.
31. 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.
32. G. Zhao, T. Wang, Y. Jin, C. Lang, Y. Li, H. Ling, The Cascaded Forward algorithm for neural network training, Pattern Recognition **161**, 111292, 2025.
33. Wang, L., Ye, J., Zhao, Y., Wu, W., Li, A., Song, S. L., ... & Kraska, T. (2018, February). Superneurons: Dynamic GPU memory management for training deep neural networks. In Proceedings of the 23rd ACM SIGPLAN symposium on principles and practice of parallel programming (pp. 41-53).
34. M. Zhang, K. Hibi, J. Inoue, GPU-accelerated artificial neural network potential for molecular dynamics simulation, Computer Physics Communications **285**, 108655, 2023.
35. Aribowo, W., Abdullayev, V. H., Oliva, D., Mahardhika, V., & Mzili, T. (2025). Metaheuristic Algorithm for Smart Grids Problems: A Brief Review. International Journal of Robotics and Control Systems, 5(6), 3085-3102.
36. Sabo, A., Bawa, M., Yakubu, Y., Ngyarmunta, A. A., Aliyu, Y., Musa, A., & Katun, M. (2025). PID controller tuning for an AVR system using Particle Swarm Optimisation Techniques and Genetic Algorithm Techniques; A comparison based approach. Vokasi Unesa Bulletin of Engineering, Technology and Applied Science, 2(2), 270-280.
37. Kim, J. K., Lee, M. Y., Kim, J. Y., Kim, B. J., & Lee, J. H. (2016, October). An efficient pruning and weight sharing method for neural network. In 2016 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia) (pp. 1-2). IEEE.
38. Nowlan, S. J., & Hinton, G. E. (2018). Simplifying neural networks by soft weight sharing. In The mathematics of generalization (pp. 373-394). CRC Press.
39. S.J. Hanson and L.Y. Pratt, Comparing biases for minimal network construction with back propagation, In D.S. Touretzky (Ed.), Advances in Neural Information Processing Systems, Volume 1, pp. 177-185, San Mateo, CA: Morgan Kaufmann, 1989.
40. M. Augasta and T. Kathirvalavakumar, Pruning algorithms of neural networks — a comparative study, Central European Journal of Computer Science, 2003.
41. Lutz Prechelt, Automatic early stopping using cross validation: quantifying the criteria, Neural Networks **11**, pp. 761-767, 1998.

42. X. Wu and J. Liu, A New Early Stopping Algorithm for Improving Neural Network Generalization, 2009 Second International Conference on Intelligent Computation Technology and Automation, Changsha, Hunan, 2009, pp. 15-18.
43. Tessier, H., Gripon, V., Léonardon, M., Arzel, M., Hannagan, T., & Bertrand, D. (2022). Rethinking weight decay for efficient neural network pruning. Journal of Imaging, 8(3), 64.
44. M. Carvalho and T. B. Ludermir, Particle Swarm Optimization of Feed-Forward Neural Networks with Weight Decay, 2006 Sixth International Conference on Hybrid Intelligent Systems (HIS'06), Rio de Janeiro, Brazil, 2006, pp. 5-5.
45. 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.
46. P.G. Benardos, G.C. Vosniakos, Optimizing feedforward artificial neural network architecture, Engineering Applications of Artificial Intelligence **20**, pp. 365-382, 2007.
47. B.A. Garro, R.A. Vázquez, Designing Artificial Neural Networks Using Particle Swarm Optimization Algorithms, Computational Intelligence and Neuroscience, 369298, 2015.
48. Siebel, N. T., & Sommer, G. (2007). Evolutionary reinforcement learning of artificial neural networks. International Journal of Hybrid Intelligent Systems, 4(3), 171-183.
49. Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, 5(2), 157–166.
50. Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS).
51. He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. ICCV.
52. L. Ingber, Very fast simulated re-annealing, Mathematical and Computer Modelling **12**, pp. 967-973, 1989.
53. Aerts, J. C., & Heuvelink, G. B. (2002). Using simulated annealing for resource allocation. International Journal of Geographical Information Science, 16(6), 571-587.
54. K. Ganesh, M. Punniyamoorthy, Optimization of continuous-time production planning using hybrid genetic algorithms-simulated annealing, Int J Adv Manuf Technol **26**, pp. 148–154, 2005.
55. El-Naggar, K. M., AlRashidi, M. R., AlHajri, M. F., & Al-Othman, A. K. (2012). Simulated annealing algorithm for photovoltaic parameters identification. Solar Energy, 86(1), 266-274.
56. Dupanloup, I., Schneider, S., & Excoffier, L. (2002). A simulated annealing approach to define the genetic structure of populations. Molecular ecology, 11(12), 2571-2581.
57. Suppapitnarm, A., Seffen, K. A., Parks, G. T., & Clarkson, P. J. (2000). A simulated annealing algorithm for multiobjective optimization. Engineering optimization, 33(1), 59-85.
58. Leite, N., Melício, F., & Rosa, A. C. (2019). A fast simulated annealing algorithm for the examination timetabling problem. Expert Systems with Applications, 122, 137-151.
59. Geng, X., Chen, Z., Yang, W., Shi, D., & Zhao, K. (2011). Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search. Applied Soft Computing, 11(4), 3680-3689.
60. Nourani, Y., & Andresen, B. (1998). A comparison of simulated annealing cooling strategies. Journal of Physics A: Mathematical and General, 31(41), 8373.
61. Karabin, M., & Stuart, S. J. (2020). Simulated annealing with adaptive cooling rates. The Journal of Chemical Physics, 153(11).
62. Anastasopoulos, N., Tsoulos, I.G., Karvounis, E. et al. Locate the Bounding Box of Neural Networks with Intervals. Neural Process Lett 52, 2241–2251 (2020).
63. M.J.D Powell, A Tolerant Algorithm for Linearly Constrained Optimization Calculations, Mathematical Programming **45**, pp. 547-566, 1989.
64. P. Kaelo, M.M. Ali, Integrated crossover rules in real coded genetic algorithms, European Journal of Operational Research **176**, pp. 60-76, 2007.
65. M. Kelly, R. Longjohn, K. Nottingham, The UCI Machine Learning Repository, https://archive.ics.uci.edu.
66. J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera. KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. Journal of Multiple-Valued Logic and Soft Computing 17, pp. 255-287, 2011.
67. Weiss, Sholom M. and Kulikowski, Casimir A., Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems, Morgan Kaufmann Publishers Inc, 1991.
68. 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.
69. J.R. Quinlan, Simplifying Decision Trees. International Journal of Man-Machine Studies **27**, pp. 221-234, 1987.
70. T. Shultz, D. Mareschal, W. Schmidt, Modeling Cognitive Development on Balance Scale Phenomena, Machine Learning **16**, pp. 59-88, 1994.

71. 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.
72. R. Setiono , W.K. Leow, FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks, Applied Intelligence **12**, pp. 15-25, 2000.
73. G. Demiroz, H.A. Govenir, N. Ilter, Learning Differential Diagnosis of Eryhemato-Squamous Diseases using Voting Feature Intervals, Artificial Intelligence in Medicine. **13**, pp. 147–165, 1998.
74. 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.
75. 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.
76. I. Kononenko, E. Šimec, M. Robnik-Šikonja, Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF, Applied Intelligence **7**, pp. 39–55, 1997
77. 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.
78. J.G. Dy , C.E. Brodley, Feature Selection for Unsupervised Learning, The Journal of Machine Learning Research **5**, pp 845–889, 2004.
79. S. J. Perantonis, V. Virvilis, Input Feature Extraction for Multilayered Perceptrons Using Supervised Principal Component Analysis, Neural Processing Letters **10**, pp 243–252, 1999.
80. J. Garcke, M. Griebel, Classification with sparse grids using simplicial basis functions, Intell. Data Anal. **6**, pp. 483-502, 2002.
81. J. Mcdermott, R.S. Forsyth, Diagnosing a disorder in a classification benchmark, Pattern Recognition Letters **73**, pp. 41-43, 2016.
82. 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.
83. 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.
84. 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.
85. 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.
86. 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.
87. 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.
88. 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.
89. T. Hastie, R. Tibshirani, Non-parametric logistic and proportional odds regression, JRSS-C (Applied Statistics) **36**, pp. 260–276, 1987.
90. M. Dash, H. Liu, P. Scheuermann, K. L. Tan, Fast hierarchical clustering and its validation, Data & Knowledge Engineering **44**, pp 109–138, 2003.
91. 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.
92. 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.
93. 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.
94. 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.
95. 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.
96. P. Zhong, M. Fukushima, Regularized nonsmooth Newton method for multi-class support vector machines, Optimization Methods and Software **22**, pp. 225-236, 2007.

97. 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.
98. 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
99. M. Koivisto, K. Sood, Exact Bayesian Structure Discovery in Bayesian Networks, The Journal of Machine Learning Research **5**, pp. 549–573, 2004.
100. 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
101. 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).
102. I.Cheng Yeh, Modeling of strength of high performance concrete using artificial neural networks, Cement and Concrete Research. **28**, pp. 1797-1808, 1998.
103. Friedman, J. (1991): Multivariate Adaptative Regression Splines. Annals of Statistics, 19:1, 1--141.
104. D. Harrison and D.L. Rubinfeld, Hedonic prices and the demand for clean ai, J. Environ. Economics & Management **5**, pp. 81-102, 1978.
105. Tsoulos, I. G., Charilogis, V., Kyrou, G., Stavrou, V. N., & Tzallas, A. (2025). OPTIMUS: A Multidimensional Global Optimization Package. Journal of Open Source Software, 10(108), 7584.
106. J. Park and I. W. Sandberg, Universal Approximation Using Radial-Basis-Function Networks, Neural Computation **3**, pp. 246-257, 1991.
107. G.A. Montazer, D. Giveki, M. Karami, H. Rastegar, Radial basis function neural networks: A review. Comput. Rev. J **1**, pp. 52-74, 2018.
108. K. O. Stanley, R. Miikkulainen, Evolving Neural Networks through Augmenting Topologies, Evolutionary Computation **10**, pp. 99-127, 2002.
109. Zhu, V., Lu, Y., & Li, Q. (2006). MW-OBS: An improved pruning method for topology design of neural networks. Tsinghua Science and Technology, 11(4), 307-312.
110. Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS), Vol. 9, pp. 249–256. PMLR.
111. He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1026–1034.
112. LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278–2324.
113. Lou, Z., & Reinitz, J. (2016). Parallel simulated annealing using an adaptive resampling interval. Parallel computing, 53, 23-31.
114. Ye, Z., Xiao, K., Ge, Y., & Deng, Y. (2018). Applying simulated annealing and parallel computing to the mobile sequential recommendation. IEEE Transactions on Knowledge and Data Engineering, 31(2), 243-256.