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. The present work proposes the introduction of an additional stage in the training of artificial neural networks before the application of the optimization method, in which stage a method based on simulated annealing is used to effectively identify the optimal value interval for the parameters of the artificial neural network. Subsequently, the optimization method will adjust the parameters of the artificial neural network within the optimal value interval of the first stage. 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

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:

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

Received:
Revised:
Accepted:
Published:

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:

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

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:

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

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 as:

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

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].

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 [35,36], pruning methods [37,38], early stopping techniques [39,40], weight decaying [41,42] 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 [43,44] or the use of the PSO technique for the efficient creation of the architecture of artificial neural networks [45]. Another method that was proposed to handle the overfitting problem is the incorporation of reinforcement learning as suggested by Siebel at al. [46].

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.

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 [48], that can be used in a magnitude of problems, such as resource allocation [49], portfolio problems [50], energy problems [51], biology [52], multi objective optimization [53], the timetabling problem [54], the traveling salesman problem [55] 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 [56,57].

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. [58] 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.

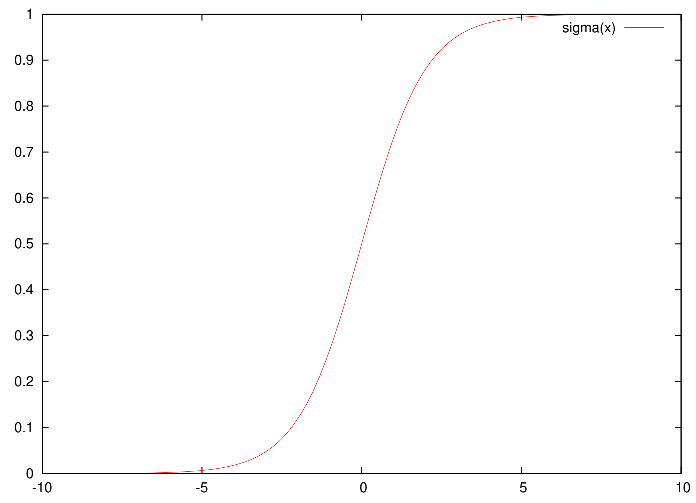


Figure 1. Plot of the sigmoid function $\sigma(x)$ in the range $[-5,5]$.

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
```

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 $i = 1, \ldots, n$ 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 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 $p_s \le 1$
      iv. p_m the mutation rate, with $p_m \le 1$
   (b) **Initialize** the chromosomes $g_i,\ i = 1, \ldots, N_c$ 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** $i = 1, \ldots, N_c$ **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 $(\tilde{z}, \tilde{w})$ 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:

$$\begin{aligned} \tilde{z}_i &= r_i z_i + (1 - r_i) w_i,\ i = 1, \ldots, n \\ \tilde{w}_i &= r_i w_i + (1 - r_i) z_i,\ i = 1, \ldots, n \end{aligned} \quad (6)$$

The numbers r_i are randomly selected numbers in $[-0.5, 1.5]$ [60].

(c) **Mutation:** This procedure is applied in every element of each chromosome, where a random number $r \in [0,1]$ is selected. If $r \leq p_m$ 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 [59] 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)[61]
2. The Keel website, https://sci2s.ugr.es/keel/datasets.php(accessed on 31 December 2025)[62].
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 [63].
2. **Alcohol** dataset, which is a dataset regarding experiments on alcohol consumption [64].
3. **Australian** dataset, derived from bank transactions [65].
4. **Balance** dataset [66], which was used in a series of psychological experiments.
5. **Cleveland**, which was studied in a series of papers [67,68].
6. **Circular** dataset, which is an artificial dataset with two classes.
7. **Dermatology**, that provides measurements from dermatology problems [69].
8. **Ecoli**, which is related to protein problems [70].
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 [71].
12. **Heart**, used to predict the presence of heart diseases [72].

13. **HeartAttack**, a medical dataset used in heart diseases
14. **Housevotes**, a dataset related to the Congressional voting in USA [73].
15. **Ionosphere**, related to measurements from the ionosphere and studied in a series of papers [74,75].
16. **Liverdisorder**, a medical dataset [76,77].
17. **Lymography** dataset[78].
18. **(OK)Mammographic**, which is related to breast cancer detection [79].
19. **Parkinsons**, which is related to the detection of Parkinson's disease [80,81].
20. **Pima**, related to the detection of diabetes[82].
21. **Phoneme**, a dataset that contains sound measurements.
22. **Popfailures**, which is related to measurements regarding climate [83].
23. **Regions2**, a dataset used for the detection of liver problems [84].
24. **Saheart**, which is a medical dataset concerning heart diseases[85].
25. **Segment** dataset [86].
26. **Statheart**, a dataset related to the detection of heart diseases.
27. **Spiral**, an artificial dataset with two classes.
28. **(OK)Student**, which is a dataset regarding experiments in schools [87].
29. **Transfusion** dataset [88].
30. **Wdbc**, used to detect the breast cancer [89,90].
31. **Wine**, a dataset used to detect the quality of wines [91,92].
32. **EEG**, that contains EEG measurements [93,94]. 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 [95] .

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

1. **Abalone**, regarding the detection of the age of abalones [96].
2. **Airfoil**, a dataset derived from NASA [97].
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 [98].
8. **DEE**, a dataset used to predict the prices of electricity.
9. **Friedman** dataset[99].
10. **FY,** related to fruit flies.
11. **HO**, a dataset derived from the STATLIB repository.
12. **Housing**, used to predict the prices of houses [100].
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 [101] 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
I_w	Weight parameter	10.0
T_0	Initial temperature	3.0
r_t	Decrease factor for temperature	0.95
N_{eps}	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

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 [102,103] with $H = 10$ hidden nodes.
6. The column NEAT denotes the incorporation of the NEAT method (NeuroEvolution of Augmenting Topologies) [104].
7. The column PRUNE denotes the application of the OBS pruning method [105].
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.

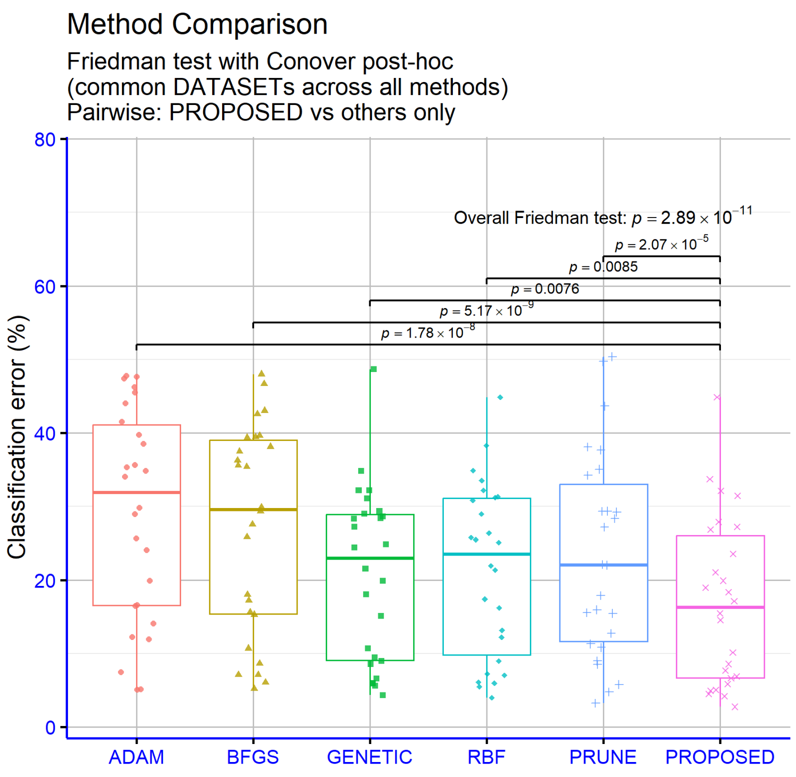


Figure 2. Statistical significance levels (Friedman and post-hoc) for classification experiments across learning models

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 3 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=3.35×10^{-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 3 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.

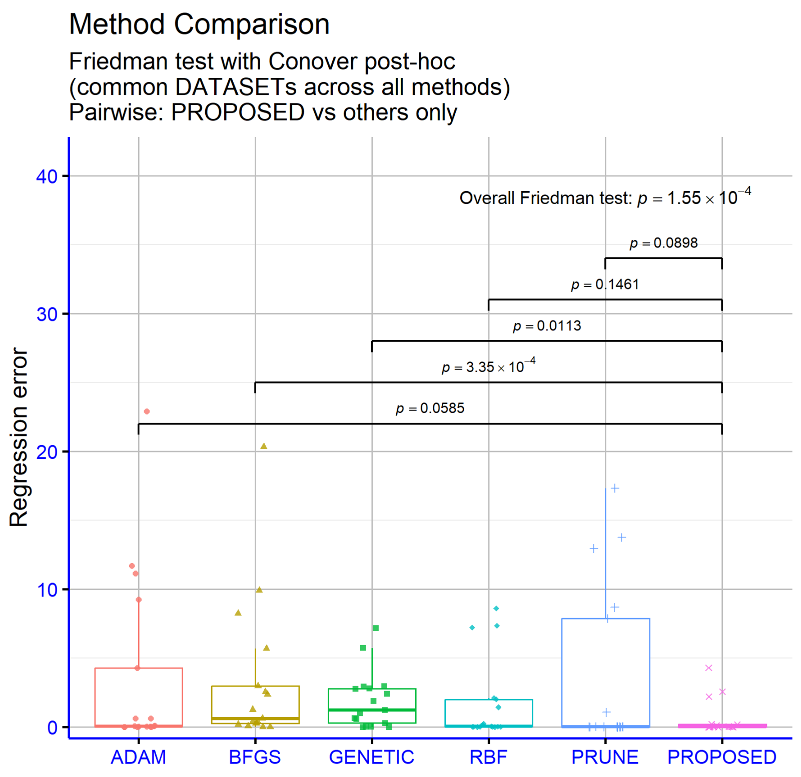


Figure 3. Statistical significance levels (Friedman and post-hoc) for regression experiments across learning models

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 4 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.

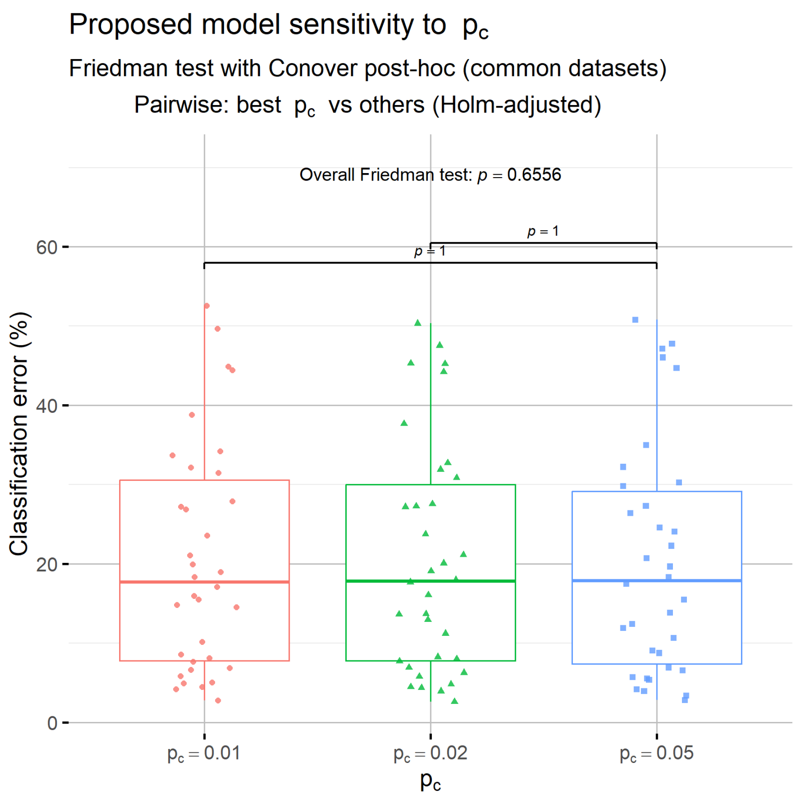


Figure 4. Statistical comparison of p_c settings for the proposed model on classification datasets

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 $\approx$ 5.27), followed closely by p_c=0.01 ($\approx$ 5.33), where asp_c=0.05 is higher ($\approx$ 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 5 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.

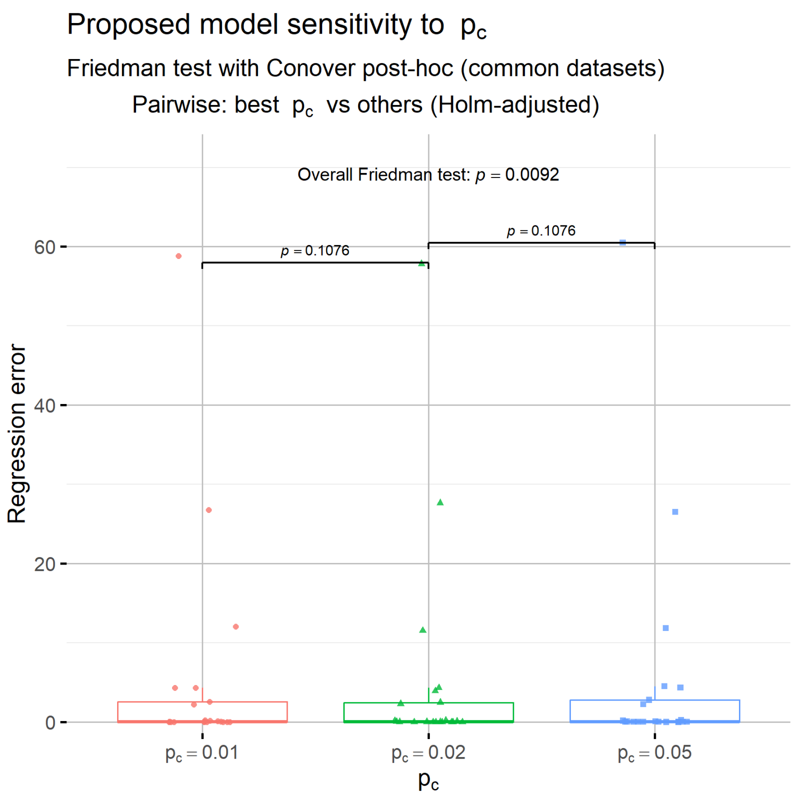


Figure 5. Statistical comparison of p_c settings for the proposed model on regression datasets

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.

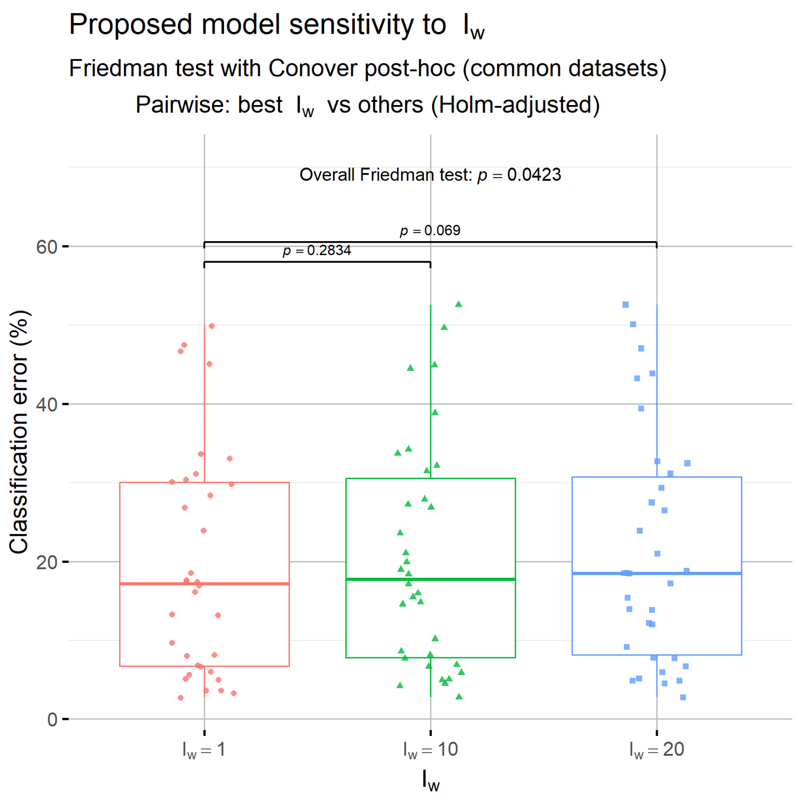
Table 6 examines the sensitivity of the proposed classifier to the parameter I_w using three settings (1, 10, 20) across 34 classification datasets, where lower error rates indicate better performance. The aggregate trend is monotonic and unfavorable as I_w increases: the average error rises from 19.45% 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 6 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.



I_w

Figure 6. 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 7 indicates that varying the Iw 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.

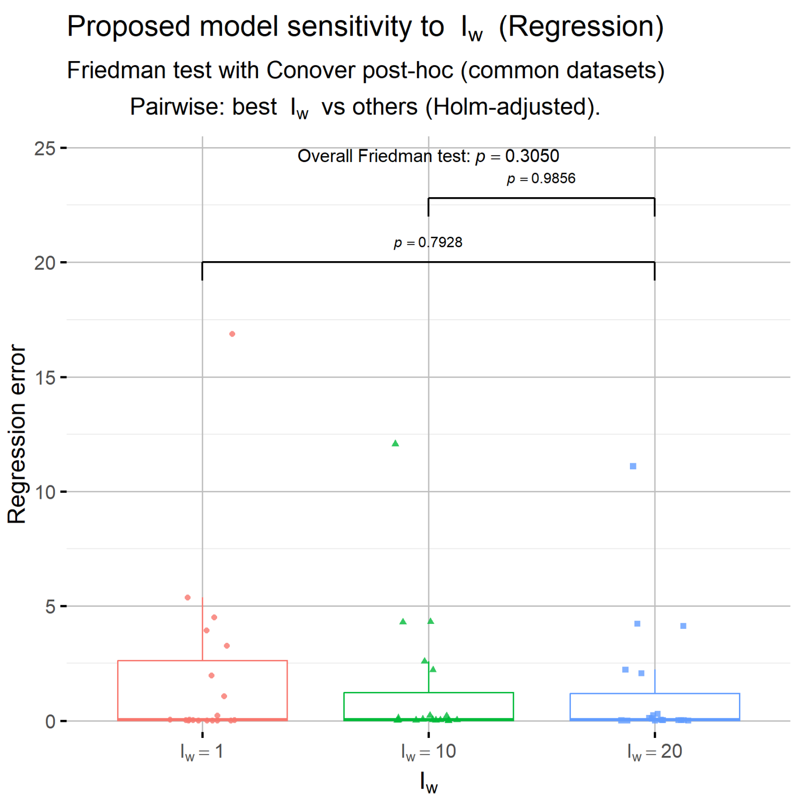


Figure 7. Statistical comparison of I_w settings for the proposed model on regression datasets

4. Conclusions

The proposed training procedure improves generalization by introducing, prior to the main optimization, an additional stage that identifies an appropriate range of values for the network weights. This stage is based on a Simulated Annealing variant that adjusts the admissible weight bounds in order to reduce the likelihood of getting trapped in local minima and to limit overfitting. The final training is then performed inside these bounds using a genetic optimization stage followed by a local BFGS refinement.

In the classification experiments, the proposed machine-learning model achieves the best mean performance, with an average classification error of 20.57, and with clear separation from all competing approaches. The advantage is not confined to isolated datasets, but emerges as a consistent pattern across a large portion of the benchmark suite, while also being accompanied by reduced dispersion of errors, which is compatible with more stable behavior on heterogeneous classification tasks.

The statistical analysis supports these observations. For classification, the overall Friedman test reports p=2.89×10^{-11}, thus very strongly rejecting the hypothesis of equivalent performance across methods. The post-hoc comparisons indicate statistically significant differences in favor of the proposed machine-learning model against all main baselines, with particularly strong evidence against ADAM, BFGS, and PRUNE, and highly significant differences against GENETIC and RBF.

In the regression experiments, the proposed machine-learning model attains the lowest mean absolute error, equal to 5.33, and the overall Friedman test remains significant with p=1.55×10^{-4}, indicating that genuine performance differences exist. However, the pairwise picture is more nuanced. The strongest statistical separation is observed primarily against BFGS with p=3.35×10^{-4}, and a weaker but still significant difference is observed against GENETIC with p=0.0113, whereas differences against ADAM, RBF, and PRUNE are not supported at the 0.05 level.

The sensitivity studies suggest that variations of the p_c parameter do not yield measurable statistical differences, both for classification and regression, indicating that this setting is not critical within the tested range. In contrast, the I_w parameter in classification exhibits a marginal overall effect, with an overall p=0.0423, but without clear confirmation through simple pairwise comparisons. For regression, I_w shows no statistically detectable impact, with an overall p=0.3050 and non-significant pairwise tests.

Several directions follow naturally for future work. First, the generality of adaptive weight bounds should be assessed under alternative final optimizers and across a broader set of architectures, including deeper networks and different activation functions. 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. Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., & Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. Heliyon, 4(11).
2. Suryadevara, S., & Yanamala, A. K. Y. (2021). A Comprehensive Overview of Artificial Neural Networks: Evolution, Architectures, and Applications. Revista de Inteligencia Artificial en Medicina, 12(1), 51-76.
3. 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. M. Egmont-Petersen, D. de Ridder, H. Handels, Image processing with neural networks—a review, Pattern Recognition **35**, pp. 2279-2301, 2002.
8. G.Peter Zhang, Time series forecasting using a hybrid ARIMA and neural network model, Neurocomputing **50**, pp. 159-175, 2003.
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. K-Su Oh, K. Jung, GPU implementation of neural networks, Pattern Recognition **37**, pp. 1311-1314, 2004.
34. M. Zhang, K. Hibi, J. Inoue, GPU-accelerated artificial neural network potential for molecular dynamics simulation, Computer Physics Communications **285**, 108655, 2023.
35. S.J. Nowlan and G.E. Hinton, Simplifying neural networks by soft weight sharing, Neural Computation 4, pp. 473-493, 1992.
36. Nowlan, S. J., & Hinton, G. E. (2018). Simplifying neural networks by soft weight sharing. In The mathematics of generalization (pp. 373-394). CRC Press.
37. 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.
38. M. Augasta and T. Kathirvalavakumar, Pruning algorithms of neural networks — a comparative study, Central European Journal of Computer Science, 2003.
39. Lutz Prechelt, Automatic early stopping using cross validation: quantifying the criteria, Neural Networks **11**, pp. 761-767, 1998.

40. 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.
41. N. K. Treadgold and T. D. Gedeon, Simulated annealing and weight decay in adaptive learning: the SARPROP algorithm,IEEE Transactions on Neural Networks **9**, pp. 662-668, 1998.
42. 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.
43. 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.
44. P.G. Benardos, G.C. Vosniakos, Optimizing feedforward artificial neural network architecture, Engineering Applications of Artificial Intelligence **20**, pp. 365-382, 2007.
45. B.A. Garro, R.A. Vázquez, Designing Artificial Neural Networks Using Particle Swarm Optimization Algorithms, Computational Intelligence and Neuroscience, 369298, 2015.
46. Siebel, N. T., & Sommer, G. (2007). Evolutionary reinforcement learning of artificial neural networks. International Journal of Hybrid Intelligent Systems, 4(3), 171-183.
47. A.A. Huqqani, E. Schikuta, S. Ye Peng Chen, Multicore and GPU Parallelization of Neural Networks for Face Recognition, Procedia Computer Science **18**, pp. 349-358, 2013.
48. L. Ingber, Very fast simulated re-annealing, Mathematical and Computer Modelling **12**, pp. 967-973, 1989.
49. Aerts, J. C., & Heuvelink, G. B. (2002). Using simulated annealing for resource allocation. International Journal of Geographical Information Science, 16(6), 571-587.
50. 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.
51. 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.
52. Dupanloup, I., Schneider, S., & Excoffier, L. (2002). A simulated annealing approach to define the genetic structure of populations. Molecular ecology, 11(12), 2571-2581.
53. 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.
54. 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.
55. 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.
56. Nourani, Y., & Andresen, B. (1998). A comparison of simulated annealing cooling strategies. Journal of Physics A: Mathematical and General, 31(41), 8373.
57. Karabin, M., & Stuart, S. J. (2020). Simulated annealing with adaptive cooling rates. The Journal of Chemical Physics, 153(11).
58. 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).
59. M.J.D Powell, A Tolerant Algorithm for Linearly Constrained Optimization Calculations, Mathematical Programming **45**, pp. 547-566, 1989.
60. P. Kaelo, M.M. Ali, Integrated crossover rules in real coded genetic algorithms, European Journal of Operational Research **176**, pp. 60-76, 2007.
61. M. Kelly, R. Longjohn, K. Nottingham, The UCI Machine Learning Repository, https://archive.ics.uci.edu.
62. 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.
63. 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.
64. 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.
65. J.R. Quinlan, Simplifying Decision Trees. International Journal of Man-Machine Studies **27**, pp. 221-234, 1987.
66. T. Shultz, D. Mareschal, W. Schmidt, Modeling Cognitive Development on Balance Scale Phenomena, Machine Learning **16**, pp. 59-88, 1994.
67. 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.
68. R. Setiono , W.K. Leow, FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks, Applied Intelligence **12**, pp. 15-25, 2000.

69. G. Demiroz, H.A. Govenir, N. Ilter, Learning Differential Diagnosis of Eryhemato-Squamous Diseases using Voting Feature Intervals, Artificial Intelligence in Medicine. **13**, pp. 147–165, 1998.
70. P. Horton, K.Nakai, A Probabilistic Classification System for Predicting the Cellular Localization Sites of Proteins, In: Proceedings of International Conference on Intelligent Systems for Molecular Biology **4**, pp. 109-15, 1996.
71. B. Hayes-Roth, B., F. Hayes-Roth. Concept learning and the recognition and classification of exemplars. Journal of Verbal Learning and Verbal Behavior **16**, pp. 321-338, 1977.
72. I. Kononenko, E. Šimec, M. Robnik-Šikonja, Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF, Applied Intelligence **7**, pp. 39–55, 1997
73. 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.
74. J.G. Dy , C.E. Brodley, Feature Selection for Unsupervised Learning, The Journal of Machine Learning Research **5**, pp 845–889, 2004.
75. S. J. Perantonis, V. Virvilis, Input Feature Extraction for Multilayered Perceptrons Using Supervised Principal Component Analysis, Neural Processing Letters **10**, pp 243–252, 1999.
76. J. Garcke, M. Griebel, Classification with sparse grids using simplicial basis functions, Intell. Data Anal. **6**, pp. 483-502, 2002.
77. J. Mcdermott, R.S. Forsyth, Diagnosing a disorder in a classification benchmark, Pattern Recognition Letters **73**, pp. 41-43, 2016.
78. 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.
79. 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.
80. 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.
81. 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.
82. 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.
83. 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.
84. 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.
85. T. Hastie, R. Tibshirani, Non-parametric logistic and proportional odds regression, JRSS-C (Applied Statistics) **36**, pp. 260–276, 1987.
86. M. Dash, H. Liu, P. Scheuermann, K. L. Tan, Fast hierarchical clustering and its validation, Data & Knowledge Engineering **44**, pp 109–138, 2003.
87. 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.
88. 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.
89. 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.
90. 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.
91. 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.
92. P. Zhong, M. Fukushima, Regularized nonsmooth Newton method for multi-class support vector machines, Optimization Methods and Software **22**, pp. 225-236, 2007.
93. 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.

94. 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
95. M. Koivisto, K. Sood, Exact Bayesian Structure Discovery in Bayesian Networks, The Journal of Machine Learning Research **5**, pp. 549–573, 2004.
96. 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
97. 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).
98. I.Cheng Yeh, Modeling of strength of high performance concrete using artificial neural networks, Cement and Concrete Research. **28**, pp. 1797-1808, 1998.
99. Friedman, J. (1991): Multivariate Adaptative Regression Splines. Annals of Statistics, 19:1, 1--141.
100. D. Harrison and D.L. Rubinfeld, Hedonic prices and the demand for clean ai, J. Environ. Economics & Management **5**, pp. 81-102, 1978.
101. 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.
102. J. Park and I. W. Sandberg, Universal Approximation Using Radial-Basis-Function Networks, Neural Computation **3**, pp. 246-257, 1991.
103. G.A. Montazer, D. Giveki, M. Karami, H. Rastegar, Radial basis function neural networks: A review. Comput. Rev. J **1**, pp. 52-74, 2018.
104. K. O. Stanley, R. Miikkulainen, Evolving Neural Networks through Augmenting Topologies, Evolutionary Computation **10**, pp. 99-127, 2002.
105. 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.