MDPI

Article

Combining constructed artificial neural networks with parameter constraint techniques to achieve better generalization properties

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

[1] Department of Informatics and Telecommunications, University of Ioannina, Greece;itsoulos@uoi.gr
[2] Department of Informatics and Telecommunications, University of Ioannina, Greece; v.charilog@uoi.gr
[3] Department of Engineering Informatics and Telecommunications, University of Western Macedonia, 50100 Kozani, Greece; tsalikakis@gmail.com
* Correspondence: itsoulos@uoi.gr

Abstract: A machine learning technology that has been used in a wide range of everyday problems is the recently proposed technique of constructing artificial neural networks with the assistance of Grammatical Evolution. This technique was utilized in problems from the fields of physics, chemistry, medicine, etc., in which it had excellent performance compared to other machine learning models. A key feature of these machine learning models is that they can both construct the structure of an artificial neural network efficiently and identify optimal values of the parameters of this network at the same time. However, in many cases the optimization process can become trapped in local minima of the error function, resulting in the construction of the artificial neural network not being able to be completed satisfactorily, which will lead to poor results in terms of test error. In this work, the combination of the artificial neural network construction method with an adapted genetic algorithm is proposed, which preserves the structure of the neural network that has been constructed and, in addition, appropriately adapts the parameters of the artificial neural network in order to avoid overfitting problems. The proposed method was applied on a wide series of datasets from real world problems that include classification and regression problems and it was compared against traditional machine learning models and the results are reported.

Keywords: Grammatical Evolution; Genetic Programming; Neural networks; Local Optimization

1. Introduction

A basic machine learning technique with a wide range of applications in data classification and regression problems is artificial neural networks [1,2]. Artificial neural networks are parametric machine learning model, in which learning is achieved by effectively adjusting their parameters through any optimization technique. The optimization procedure minimizes the so - called training error of an artificial neural network and it is defined as:

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

In this equation the function $N\left(\overrightarrow{x},\overrightarrow{w}\right)$ represents the artificial neural network which is applied on a vector $\overrightarrow{x}$ and the vector $\overrightarrow{w}$ denotes the parameter vector of the neural network. The set $\left(\overrightarrow{x_i},y_i\right),\ i = 1,...,M$ represents the training set of the objective problem and the values y_i are the expected outputs for each pattern $\overrightarrow{x_i}$.

Artificial neural networks have been applied in a wide series of problems appeared in real - world problems, such as image processing [3], time series forecasting [4], credit card analysis [5], problems derived from physics [6,7] etc. Due to the widespread use of these machine learning models, a number of techniques have been proposed to minimize the equation 1, such as the Back Propagation algorithm [8,9], the RPROP algorithm [10, 11], the ADAM optimization method [12] etc. Also, recently a series of more advanced

Citation: Tsoulos, I.G.; Charilogis, V.; Tsalikakis D. Combining constructed artificial neural networks with parameter constraint techniques to achieve better generalization properties. *Journal Not Specified* **2024**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

global optimization methods have been proposed to tackle the training of neural networks. Among them one can locate the incorporation of Genetic Algorithms [13], the usage of the Particle Swarm Optimization (PSO) method [14], the Simulated Annealing method [15], the Differential Evolution technique [16], the Artificial Bee Colony (ABC) method [17] etc. Furthermore, Sexton et al suggested the usage of the tabu search algorithm for optimal neural network training [18], Zhang et al proposed a hybrid algorithm that incorporated the PSO method and the Back Propagation algorithm to efficient train artificial neural networks [19]. Also, recently Zhao et al introduced a new Cascaded Forward Algorithm to train artificial neural networks [20]. Furthermore, due to the rapid spread of the use of parallel computing techniques, a series of computational techniques have emerged that exploit parallel computing structures for faster training of artificial neural networks [21,22].

However, the above techniques, although extremely effective, nevertheless have a number of problems such as, for example, trapping in local minima of the error function or the phenomenon of overifitting, where the artificial neural network exhibits reduced performance when applied to data that was not present during the training process. The overfitting problem has been studied by many researchers that have proposed a series of methods to handle this problem, such as weight sharing [23,24], pruning [25,26], early stopping [27,28], weight decaying [29,30] etc. Also, many researchers propose as a solution to the above problem the dynamic creation of the architecture of artificial neural networks using programming techniques. For example the Genetic Algorithms were proposed to create dynamically the optimal architecture of neural networks [31,32] or the PSO method [33].

A method that was proposed relatively recently and it is based on Grammatical Evolution [34], dynamically identifies both the optimal architecture of artificial neural networks and the optimal values of its parameters [35]. This method has been applied in a series of problems in the recent literature, such as problems presented in chemistry [36], identification of the solution of differential equations [37], medical problems [38], problems related to education [39], autism screening [40] etc. A key advantage of this technique is that it can isolate from the initial features of the problem those that are most important in training the model, thus significantly reducing the required number of parameters that need to be identified.

However, the method of constructing artificial neural networks can easily get trapped in local minima of the training error since it does not have any technique to avoid them. Furthermore, although the method can get quite close to a minimum of the training error, it often does not reach it since there is no technique in the method to train the generated parameters. In this technique, it is proposed to enhance the original method of constructing artificial neural networks by periodically applying a modified genetic algorithm to randomly selected chromosomes of Grammatical Evolution. This modified genetic algorithm preserves the architecture created by the Grammatical Evolution method and effectively locates the parameters of the artificial neural network by reducing the training error. In addition, the proposed genetic algorithm through appropriate penalty factors imposed on the fitness function prevents the artificial neural network from overfitting.

The remaining of this article is organized as follows: in section 2 the proposed method and the accompanied genetic algorithm are introduced, in section 3 the experimental datasets and the series of experiments conducted are listed and discussed thoroughly and final in section 4 some conclusions are discussed.

2. Method description

This section provides a details description of the original neural network construction method and continuous with the proposed genetic algorithm and concludes with the overall algorithm.

2.1. The neural construction method

The neural construction method utilizes the technique of Grammatical Evolution to produce artificial neural networks. Grammatical Evolution is an evolutionary process where the chromosomes are vectors of positive integers. These integers represent rules from a Backs - Naur form (BNF) grammar [41] of the target language. The method was incorporated in various cases, such as data fitting [42,43], composition of music [44], video games [45,46], energy problems [47], cryptography [48], economics [49] etc. Any BNF grammar is defined as a set $G = (N, T, S, P)$ where the letters have the following definitions:

- The set N represents the non - terminal symbols of the grammar.
- The set T contains the terminal symbols of the grammar.
- The start symbol of the grammar is denoted as S.
- The production rules of the grammar are enclosed in the set P

The Grammatical Evolution production procedure initiates from the starting symbol S and following a series of steps, the method creates valid programs by replacing non-terminal symbols with the right hand of the selected production rule. The selection scheme has as:

- **Read** the next element V from the chromosome that is being processed.
- **Select** the next production rule following the equation:Rule = V mod N_R. The symbol N_R represents the total number of production rules for the under processing non – terminal symbol.

The process of producing valid programs through the Grammatical Evolution method is depicted graphically in Figure 1.

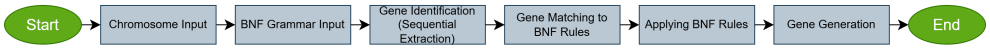

Figure 1. The Grammatical Evolution process used to produce valid programs.

The grammar used for the neural construction procedure is shown in Figure 2. The numbers shown in parentheses are the increasing numbers of the production rules for each non - terminal symbol. The constant d denotes the number of features in every pattern of the input dataset.

```
S:=<Sigval>                              (0)
<Sigval>::=<Node>                        (0)
          | <Node> + <Sigval>            (1)
<Node>::=<Number>*sig(<Sum>+<Number>) (0)
<Sum>::= <Number>*<Xlist>                (0)
          |    <Sum>+<Sum>                (1)
<Xlist>::= x1          (0)
            |    x2  (1)
            ..............
            |    xd  (d-1)
<Number>::= (<Dlist>.<Dlist>)            (0)
            |    (-<Dlist>.<Dlist>) (1)
<Dlist>::= <Digit>                 (0)
            | <Digit><Dlist> (1)
<Digit>::= 0       (0)
            |  1 (1)
            ...........
            |  9 (9)
```

Figure 2. The proposed grammar for the construction of artificial neural networks through Grammatical Evolution.

The used grammar produces artificial neural networks with the following form:

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

The term H stands for the number of processing units (weights) of the neural network. The The function $\sigma(x)$ represents the sigmoid function. The total number of parameters for this network are computed through the following equation:

$$n = (d+2)H \quad (3)$$

For example the following form:

$$N(x) = 1.9\text{sig}(10.5x_1 + 3.2x_3 + 1.4) + 2.1\text{sig}(2.2x_2 - 3.3x_3 + 3.2) \quad (4)$$

denotes a produced neural network for a problem with 3 inputs (x_1, x_2, x_3) and the number of processing nodes is $H = 2$. The neural network produced can be shown graphically in Figure 3.

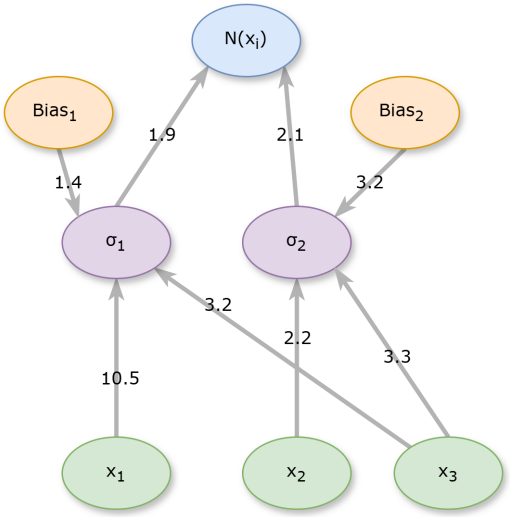

Figure 3. An example of a produced neural network.

2.2. The used genetic algorithm

In the original method of constructing artificial neural networks, it is proposed in this work to introduce the concept of local search, through the periodic application of a genetic algorithm which should maintain the structure of the neural network constructed by the original method. Additionally, a second goal of this genetic algorithm should be to avoid the problem of overfitting that could arise from simply applying a local optimization method to the previous artificial neural network. For the first goal of the modified genetic algorithm consider the example neural network shown before:

$$N(x) = 1.9\text{sig}(10.5x_1 + 3.2x_3 + 1.4) + 2.1\text{sig}(2.2x_2 - 3.3x_3 + 3.2) \quad (5)$$

The weight vector $\vec{w}$ for this neural network would be

$$\vec{w} = [1.9, 10.5, 0.0, 3.2, 1.4, 2.1, 0.0, 2.2, -3.3, 3.2] \quad (6)$$

In order to protect the structure of this artificial neural network, the modified genetic algorithm should allow changes in the parameters of this network within a value interval,

which can be considered to be the pair of vectors $\left[\overrightarrow{L}, \overrightarrow{R}\right]$. The elements for the vector $\overrightarrow{L}$ are defined as

$$L_i = -F \times |w_i|, \; i = 1, \ldots, n \tag{7}$$

where F is positive number with $F > 1$. Likewise the right bound for the parameters $\overrightarrow{R}$ is defined from the following equation:

$$R_i = F \times |w_i|, i = 1, \ldots, n \tag{8}$$

For the example weight vector of equation 6 and for $F = 2$ the following vectors are used:

$$\begin{aligned} L &= [-3.8, -21.0, 0.0, -6.4, -2.8, -4.2, 0.0, -4.4, -6.6, -6.4] \\ R &= [\; 3.8, \; 21.0, 0.0, \; 6.4, \; 2.8, \; 4.2, 0.0, \; 4.4, \; 6.6, \; 6.4] \end{aligned}$$

The modified genetic algorithm should also prevent the artificial neural networks it trains from the phenomenon of overfitting, which would lead to poor results on the test dataset. For this reason a quantity derived from the publication of Anastasopoulos et al. [50] is utilized here. The sigmoid function, that is used as the activation function of neural networks is defined as:

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{9}$$

A plot for this function is shown in Figure 4.

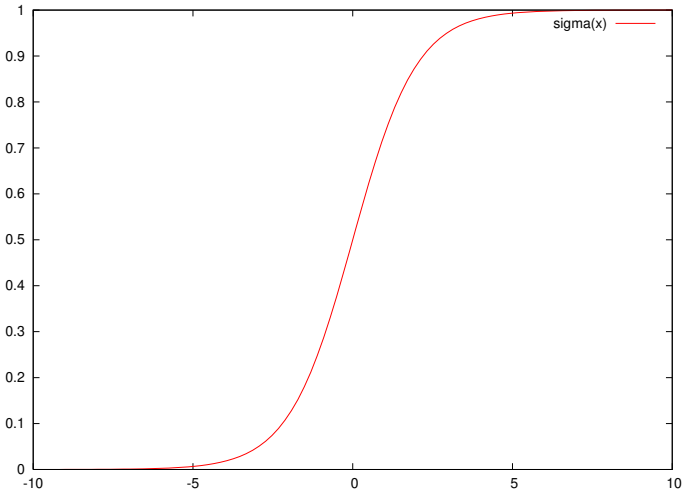

Figure 4. Plot of the sigmoid function $\sigma(x)$.

As is clear from the equation and the figure, as the value of the parameter x increases, the function tends very quickly to 1. On the other hand, the function will take values very close to 0 as the parameter x decreases. This means that the function very quickly loses the generalizing abilities it has and therefore large changes in the value of the parameter x will not cause proportional variations in the value of the sigmoid function. Therefore, the quantity $B\left(N\left(\overrightarrow{x}, \overrightarrow{w}\right), a\right)$ was introduced in that paper to measure this effect. This quantity is calculated through the process of Algorithm 1.

Algorithm 1 The algorithm used to calculate the bounding quantity for neural network $N(x, w)$.

function evalB$\left(N\left(\overrightarrow{x}, \overrightarrow{w}\right), a\right)$

1. **Inputs**: The Neural network $N\left(\overrightarrow{x}, \overrightarrow{w}\right)$ and the double precision value a, $a > 1$.
2. **Set** $s = 0$
3. **For** $i = 1..H$ **Do**
   (a) **For** $j = 1..M$ **Do**
      i. **Calculate** $v = \sum_{kT=1}^{d} w_{(d+2)i-(d+i)+k} x_{jk} + w_{(d+2)i}$
      ii. **If** $|v| > a$ **set** $s = s + 1$
   (b) **EndFor**
4. **EndFor**
5. **Return** $\frac{s}{H \star M}$

End Function

The overall proposed modified genetic algorithm is shown in Algorithm 2.

Algorithm 2 The modified Genetic Algorithm.

Function mGA$\left(\overrightarrow{L}, \overrightarrow{R}, a, \lambda\right)$

1. **Inputs**: The bound vectors $\overrightarrow{L}, \overrightarrow{R}$ and the bounding factor a and λ a positive value with $\lambda > 1$.
2. **Set** as $N_K$the number of allowed generations and as N_G the number of used chromosomes.
3. **Set** as p_S the selection rate and as p_M the mutation rate.
4. **Initialize** N_G chromosomes inside the bounding boxes $\overrightarrow{L}, \overrightarrow{R}$.
5. **Set** $k = 0$, the generation number.
6. **For** $i = 1, \ldots, N_G$
   (a) **Obtain** the corresponding neural network $N_i\left(\overrightarrow{x}, \overrightarrow{g_i}\right)$ for the chromosome g_i.
   (b) **Set** $e_i = \sum_{j=1}^{M} \left(N_i\left(\overrightarrow{x}_j, \overrightarrow{w_i}\right) - y_j\right)^2$
   (c) **Set** $B_i = \text{eval}\left(N_i\left(\overrightarrow{x}, \overrightarrow{g_i}\right), a\right)$ using the algorithm 1.
   (d) **Set** $f_i = e_i \times \left(1 + \lambda B_i^2\right)$ as the fitness value of chromosome g_i
7. **End For**
8. **Select** the best $(1 - p_s) \times N_G$ chromosomes, that will be copied intact to the next generation. The remaining will be substituted by individuals produced by crossover and mutation.
9. **Set** $k = k + 1$
10. **If** $k \leq N_K$ **goto** step 6.

End function

2.3. The overall algorithm

The overall algorithm uses the procedures presented previously to achieve greater accuracy in calculations as well as to avoid overfitting phenomena. The steps of the overall algorithm have as follows:

1. **Initialization**.
   (a) **Set** as N_C the number of chromosomes for the Grammatical Evolution procedure and as N_G the maximum number of allowed generations.
   (b) **Set** as p_S the selection rate and as p_M the mutation rate.
   (c) **Let** N_I be the number of chromosomes to which the modified genetic algorithm will be periodically applied.
   (d) **Let** N_T be the number of generations that will pass before applying the modified genetic algorithm to randomly selected chromosomes.

(e) **Set** the weight factor F with $F > 1$.
(f) **Set** the values N_K, a, λ used in the modified genetic algorithm.
(g) **Initialize** randomly the N_C chromosomes as sets of randomly selected integers.
(h) **Set** the generation number $k = 0$

2. **Fitness Calculation**.
   (a) **For** $i = 1, \ldots, N_C$ **do**
      i. **Obtain** the chromosome g_i
      ii. **Create** the corresponding neural network $N_i(\overrightarrow{x}, \overrightarrow{w})$ using Grammatical Evolution.
      iii. **Set** the fitness value $f_i = \sum_{j=1}^{M} \left(N_i(\overrightarrow{x}_j, \overrightarrow{w}) - y_j\right)^2$
   (b) **End For**
3. **Genetic Operations**.
   (a) **Select** the best $(1 - p_s) \times N_G$ chromosomes, that will be copied intact to the next generation.
   (b) **Create** $p_S N$ chromosomes using one - point crossover. For every couple (c_1, c_2) of produced offsprings two distinct chromosomes are selected from the current population using tournament selection. An example of the one - point crossover procedure is shown graphically in Figure 5.
   (c) For every chromosome and for each element select a random number $r \leq 1$. Alter the current element when $r \leq p_M$
4. **Local search.**
   (a) **If** $k \bmod N_T = 0$ **then**
      i. **Set** $S = \left\{g_{r_1}, g_{r_2}, \ldots, g_{r_{N_I}}\right\}$ a group of N_I randomly selected chromosomes from the genetic population.
      ii. **For** every member $g \in S$ **do**
         A. **Obtain** the corresponding neural network $N_g(\overrightarrow{x}, \overrightarrow{w})$ for the chromosome g.
         B. **Create** the left bound vector $\overrightarrow{L_g}$ and the right bound vector $\overrightarrow{R_g}$ for g using the equations 7,8 respectively.
         C. **Set** $g = \text{mga}\left(\overrightarrow{L_g}, \overrightarrow{R_g}, a, \lambda\right)$ using the steps of algorithm 2.
      iii. **End For**
   (b) **Endif**
5. **Termination Check**.
   (a) **Set** $k = k + 1$
   (b) **If** $k \leq N_G$ goto **Fitness Calculation**.
6. **Application to the test set**.
   (a) **Obtain** the chromosome g^* with the lowest fitness value and create through Grammatical Evolution the corresponding neural network $N^*(\overrightarrow{x}, \overrightarrow{w})$
   (b) **Apply** the neural network $N^*(\overrightarrow{x}, \overrightarrow{w})$ and report the corresponding error value.

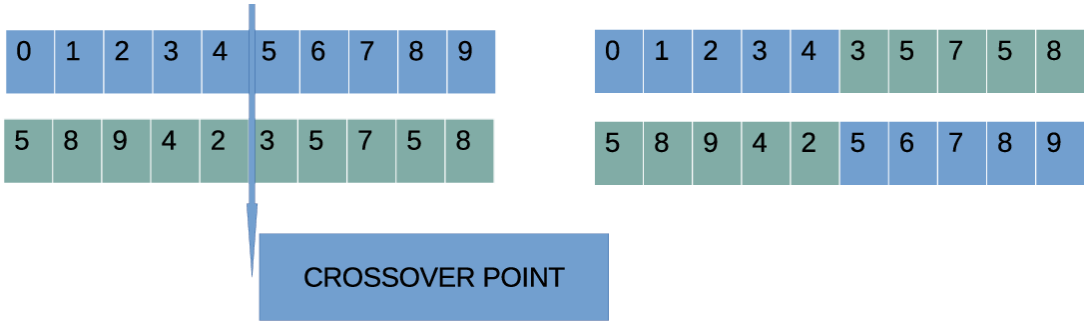

Figure 5. An example of the one - point crossover procedure.

3. Experimental results

The validation of the proposed method was performed using a wide series of classification and regression datasets, available from various sources from the Internet. These datasets were downloaded from:

1. The UCI database, https://archive.ics.uci.edu/(accessed on 22 January 2025)[51]
2. The Keel website, https://sci2s.ugr.es/keel/datasets.php(accessed on 22 January 2025)[52].
3. The Statlib URL https://lib.stat.cmu.edu/datasets/index(accessed on 22 January 2025).

3.1. Experimental datasets

The following datasets were utilized in the conducted experiments:

1. **Appendictis** which is a medical dataset [53].
2. **Alcohol**, which is dataset regarding alcohol consumption [54].
3. **Australian**, which is a dataset produced from various bank transactions [55].
4. **Balance** dataset [56], produced from various psychological experiments.
5. **Cleveland**, a medical dataset which was discussed in a series of papers [57,58].
6. **Circular** dataset, which is an artificial dataset.
7. **Dermatology**, a medical dataset for dermatology problems [59].
8. **Ecoli**, which is related to protein problems [60].
9. **Glass** dataset, that contains measurements from glass component analysis.
10. **Haberman**, a medical dataset related to breast cancer.
11. **Hayes-roth** dataset [61].
12. **Heart**, which is a dataset related to heart diseases [62].
13. **HeartAttack**, which is a medical dataset for the detection of heart diseases
14. **Housevotes**, a dataset which is related to the Congressional voting in USA [63].
15. **Ionosphere**, a dataset that contains measurements from the ionosphere [64,65].
16. **Liverdisorder**, a medical dataset that was studied thoroughly in a series of papers[66, 67].
17. **Lymography** [68].
18. **Mammographic**, which is a medical dataset used for the prediction of breast cancer [69].
19. **Parkinsons**, which is a medical dataset used for the detection of Parkinson's disease [70,71].
20. **Pima**, which is a medical dataset for the detection of diabetes[72].
21. **Phoneme**, a dataset that contains sound measurements.
22. **Popfailures**, a dataset related to experiments regarding climate [73].
23. **Regions2**, a medical dataset applied to liver problems [74].
24. **Saheart**, which is a medical dataset concerning heart diseases[75].
25. **Segment** dataset [76].
26. **Statheart**, a medical dataset related to heart diseases.

27. **Spiral**, an artificial dataset with two classes.
28. **Student**, which is a dataset regarding experiments in schools [77].
29. **Transfusion**, which is a medical dataset [78].
30. **Wdbc**, which is a medical dataset regarding breast cancer [79,80].
31. **Wine**, a dataset regarding measurements about the quality of wines [81,82].
32. **EEG**, which is dataset regardingEEG recordings [83,84]. From this dataset the following cases were used: Z_F_S, ZO_NF_S, ZONF_S and Z_O_N_F_S.
33. **Zoo**, which is a dataset regarding animal classification [85] .

Moreover a series of regression datasets was adopted in the conducted experiments. The list with the regression datasets has as follows:

1. **Abalone**, which is a dataset about the age of abalones [86].
2. **Airfoil**, a dataset founded in NASA [87].
3. **Auto**, a dataset related to the consumption of fuels from cars.
4. **BK**, which is used to predict the points scored in basketball games.
5. **BL**, a dataset that contains measurements from electricity experiments.
6. **Baseball**, which is a dataset used to predict the income of baseball players.
7. **Concrete**, which is a civil engineering dataset [88].
8. **DEE**, a dataset that is used to predict the price of electricity.
9. **Friedman**, which is an artificial dataset[89].
10. **FY,** which is a dataset regarding the longevity of fruit flies.
11. **HO**, a dataset located in the STATLIB repository.
12. **Housing**, regarding the price of houses [90].
13. **Laser**, which contains measurements from various physics experiments.
14. **LW**, a dataset regarding the weight of babes.
15. **Mortgage**, a dataset that contains measurements from the economy of USA.
16. **PL** dataset, located in the STALIB repository.
17. **Plastic**, a dataset regarding problems occurred with the pressure on plastics.
18. **Quake**, a dataset regarding the measurements of earthquakes.
19. **SN**, a dataset related to trellising and pruning.
20. **Stock**, which is a dataset regarding stocks.
21. **Treasury**, a dataset that contains measurements from the economy of USA.

3.2. Experiments

The software used in the experiment was coded in C++ with the assistance of the freely available Optimus environment, that can be downloaded from https://github.com/itsoulos/GlobalOptimus/(accessed on 23 January 2025). Every experiments was conducted 30 times and each time different seed for the random generator was used. The experiments were validated using the ten - fold cross validation technique. The average classification error, as measured in the corresponding test set was reported for the classification datasets. This error is calculated through the following formula:

$$E_C\left(N\left(\vec{x},\vec{w}\right)\right) = 100 \times \frac{\sum_{i=1}^{N}\left(\text{class}\left(N\left(\vec{x_i},\vec{w}\right)\right)-y_i\right)}{N} \tag{10}$$

Here the test set T is a set $T = (x_i, y_i),\ i = 1,\ldots,N$. Likewise, the average regression error is reported for the regression datasets. This error can be obtained using the following equation:

$$E_R\left(N\left(\vec{x},\vec{w}\right)\right) = \frac{\sum_{i=1}^{N}\left(N\left(\vec{x_i},\vec{w}\right)-y_i\right)^2}{N} \tag{11}$$

The experiments were executed on an AMD Ryzen 5950X with 128GB of RAM and the used operating system was Debian Linux. The values for the parameters of the proposed method are shown in Table 1.

PARAMETER	MEANING	VALUE
N_C	Chromosomes	500
N_G	Maximum number of generations	500
N_K	Number of generations for the modified Genetic Algorithm	50
p_S	Selection rate	0.1
p_M	Mutation rate	0.05
N_T	Generations before local search	20
N_I	Chromosomes participating in local search	20
a	Bounding factor	10.0
F	Scale factor for the margins	2.0
λ	Value used for penalties	100.0

Table 1. The values for the parameters of the proposed method.

In the following tables that describe the experimental results the following notation is used:

1. The column DATASET represents the used dataset.
2. The column ADAM represents the incorporation of the ADAM optimization method [12] to train a neural network with $H = 10$ processing nodes.
3. The column BFGS stands for the usage of a BFGS variant of Powell [91] to train an artificial neural network with $H = 10$ processing nodes.
4. The column GENETIC represents the incorporation of a Genetic Algorithm with the same parameter set as provided in Table 1 to train a neural network with $H = 10$ processing nodes.
5. The column RBF describes the experimental results obtained by the application of a Radial Basis Function (RBF) network [92,93] with $H = 10$ hidden nodes.
6. The column NNC stands for the usage of the original neural construction method.
7. The column PROPOSED denotes the usage of the proposed method.
8. The row AVERAGE represents the average classification or regression error for all datasets in the corresponding table.

In Table 2, the performance of different machine learning algorithms on various classification datasets is shown, as measured by error rates. Each column represents a specific machine learning model, while each row corresponds to a dataset and the error rate associated with each algorithm for that particular dataset. The last row of the table presents the average error rates for each algorithm, providing an overall view of their performance. The analysis begins with the observation that the "PROPOSED" model demonstrates the lowest average error rate (21.18%) compared to all other models. This indicates that, overall, it delivers the best performance, producing the most accurate results across this set of data. In many datasets, such as APPENDICITIS, BALANCE, and ZOO, the "PROPOSED" model achieves the lowest error rates, specifically 14.30%, 7.84%, and 6.60%, respectively, showcasing its strong performance in these cases. However, it is not consistently the best, as in some datasets, like MAMMOGRAPHIC and CIRCULAR, other models exhibit slightly lower error rates.The "GENETIC" model records the highest average error rate, at 28.25%, which suggests its overall performance is the weakest across this dataset collection. However, it is noted that in certain datasets, such as CIRCULAR and HOUSEVOTES, the "GENETIC" model performs relatively well with error rates of 5.99% and 6.62%, respectively, which are comparable to those of the more effective models. This indicates that its performance may depend on the specific characteristics of each dataset. Another noteworthy algorithm is "BFGS," which has an average error rate of 35.71%. While its performance is not consistently strong, significant deviations are observed in certain datasets, such as DERMATOLOGY and SEGMENT. For instance, in DERMATOLOGY, it exhibits an error rate of 52.92%, whereas the "PROPOSED" model achieves 20.54%, highlighting the substantial sensitivity of "BFGS" to the peculiarities of this dataset. In datasets with substantial variability in error rates, such as AUSTRALIAN and STATHEART, significant differentiation between algo-

rithms is observed. In AUSTRALIAN, the "PROPOSED" model achieves the lowest error rate of 14.55%, while other algorithms exhibit error rates ranging from 32.21% to 38.13%. This difference underscores the superiority of the "PROPOSED" model in datasets that pose specific challenges to other algorithms. Similarly, in STATHEART, the "PROPOSED" model achieves one of the lowest error rates (17.93%), whereas other algorithms, such as "GENETIC," show rates as high as 27.25%.

Table 2. Experimental results using a variety of machine learning methods for the classification datasets.

DATASET	ADAM	BFGS	GENETIC	RBF	NNC	PROPOSED
APPENDICITIS	16.50%	18.00%	24.40%	12.23%	14.40%	14.30%
ALCOHOL	57.78%	41.50%	39.57%	49.32%	37.72%	35.60%
AUSTRALIAN	35.65%	38.13%	32.21%	34.89%	14.46%	14.55%
BALANCE	12.27%	8.64%	8.97%	33.53%	23.65%	7.84%
CLEVELAND	67.55%	77.55%	51.60%	67.10%	50.93%	46.41%
CIRCULAR	19.95%	6.08%	5.99%	5.98%	12.66%	6.92%
DERMATOLOGY	26.14%	52.92%	30.58%	62.34%	21.54%	20.54%
ECOLI	64.43%	69.52%	54.67%	59.48%	49.88%	48.82%
GLASS	61.38%	54.67%	52.86%	50.46%	56.09%	53.52%
HABERMAN	29.00%	29.34%	28.66%	25.10%	27.53%	26.80%
HAYES-ROTH	59.70%	37.33%	56.18%	64.36%	33.69%	31.00%
HEART	38.53%	39.44%	28.34%	31.20%	15.67%	15.45%
HEARTATTACK	45.55%	46.67%	29.03%	29.00%	20.87%	21.77%
HOUSEVOTES	7.48%	7.13%	6.62%	6.13%	3.17%	3.78%
IONOSPHERE	16.64%	15.29%	15.14%	16.22%	11.29%	11.94%
LIVERDISORDER	41.53%	42.59%	31.11%	30.84%	32.35%	31.32%
LYMOGRAPHY	39.79%	35.43%	28.42%	25.50%	25.29%	23.72%
MAMMOGRAPHIC	46.25%	17.24%	19.88%	21.38%	17.62%	16.74%
PARKINSONS	24.06%	27.58%	18.05%	17.41%	12.74%	12.63%
PHONEME	29.43%	15.58%	15.55%	23.32%	22.50%	21.52%
PIMA	34.85%	35.59%	32.19%	25.78%	28.07%	23.34%
POPFAILURES	5.18%	5.24%	5.94%	7.04%	6.98%	5.72%
REGIONS2	29.85%	36.28%	29.39%	38.29%	26.18%	23.81%
SAHEART	34.04%	37.48%	34.86%	32.19%	29.80%	28.04%
SEGMENT	49.75%	68.97%	57.72%	59.68%	53.50%	48.20%
SPIRAL	47.67%	47.99%	48.66%	44.87%	48.01%	44.95%
STATHEART	44.04%	39.65%	27.25%	31.36%	18.08%	17.93%
STUDENT	5.13%	7.14%	5.61%	5.49%	6.70%	4.05%
TRANSFUSION	25.68%	25.84%	24.87%	26.41%	25.77%	23.16%
WDBC	35.35%	29.91%	8.56%	7.27%	7.36%	4.95%
WINE	29.40%	59.71%	19.20%	31.41%	13.59%	9.94%
Z_F_S	47.81%	39.37%	10.73%	13.16%	14.53%	7.97%
Z_O_N_F_S	78.79%	65.67%	64.81%	48.70%	48.62%	39.28%
ZO_NF_S	47.43%	43.04%	21.54%	9.02%	13.54%	6.94%
ZONF_S	11.99%	15.62%	4.36%	4.03%	2.64%	2.60%
ZOO	14.13%	10.70%	9.50%	21.93%	8.70%	6.60%
AVERAGE	**36.45%**	**35.71%**	**28.25%**	**30.73%**	**24.79%**	**21.18%**

The table 3 presents the performance of various machine learning models on regression datasets, as measured by absolute error values. Each column represents a specific model, while each row corresponds to a dataset and the absolute error value associated with each algorithm for that dataset. The last row provides the average error values for each algorithm, offering a general view of their overall performance. The analysis shows that the "PROPOSED" model achieves the lowest average error value (4.28) among all

the algorithms, indicating its superior overall performance in terms of accuracy. Across multiple datasets, the "PROPOSED" model consistently achieves the smallest error values, such as in datasets like BL (0.001), MORTGAGE (0.023), and TREASURY (0.068). These results highlight its effectiveness across various regression problems. However, it is notable that in some datasets, such as FY (0.043) and HOUSING (15.47), the "PROPOSED" model does not have the lowest error values, though its performance remains competitive. On the other hand, the "BFGS" model records the highest average error value (30.29), reflecting its generally weaker performance across the datasets. Its errors are significantly higher in certain datasets, such as BASEBALL (119.63) and STOCK (302.43), indicating challenges in addressing regression tasks effectively in these cases. Despite this, "BFGS" shows competitive performance in a few instances, such as in the AIRFOIL dataset, where it achieves an error value of 0.003, close to that of the "PROPOSED" model. The "GENETIC" model has an average error value of 9.31, placing it among the mid-performing models in this analysis. It performs relatively well in datasets like MORTGAGE (2.41) and TREASURY (2.93), but struggles in others, such as BASEBALL (103.6) and PLASTIC (2.791). These results indicate variability in its performance depending on the dataset characteristics. The "ADAM" model, with an average error value of 22.46, shows moderate performance overall. It has strong results in some datasets, such as AIRFOIL (0.005) and CONCRETE (0.078), but performs poorly in others, such as STOCK (180.89) and BASEBALL (77.9). This suggests that "ADAM" is sensitive to dataset-specific features and may not generalize well across all cases. The "RBF" model achieves an average error value of 10.02, demonstrating moderately good performance across the datasets. It performs particularly well in datasets like MORTGAGE (1.45) and LASER (0.03). However, it faces challenges in datasets such as BASEBALL (93.02) and STOCK (12.23), where its error values are notably higher. The "NNC" model has an average error value of 6.29, placing it as a strong performer, second only to the "PROPOSED" model in this analysis. It shows competitive results across many datasets, such as PL (0.047) and HO (0.015), but it is slightly outperformed by "PROPOSED" in most cases, including critical datasets like BL and TREASURY. Overall, the analysis reveals the "PROPOSED" model's superiority in minimizing absolute error across diverse regression datasets, as reflected by its lowest average error. The results also underscore the variability in performance among the other algorithms, suggesting that certain models may perform better in specific contexts but lack consistent efficacy across all datasets.

Table 3. Experimental results using a variety of machine learning methods on the regression datasets.

DATASET	ADAM	BFGS	GENETIC	RBF	NNC	PROPOSED
ABALONE	4.30	5.69	7.17	7.37	5.08	4.47
AIRFOIL	0.005	0.003	0.003	0.27	0.004	0.002
AUTO	70.84	60.97	12.18	17.87	17.13	9.09
BK	0.0252	0.28	0.027	0.02	0.10	0.023
BL	0.622	2.55	5.74	0.013	1.19	0.001
BASEBALL	77.90	119.63	103.60	93.02	61.57	48.13
CONCRETE	0.078	0.066	0.0099	0.011	0.008	0.005
DEE	0.63	2.36	1.013	0.17	0.26	0.22
FRIEDMAN	22.90	1.263	1.249	7.23	6.29	5.34
FY	0.038	0.19	0.65	0.041	0.11	0.043
HO	0.035	0.62	2.78	0.03	0.015	0.016
HOUSING	80.99	97.38	43.26	57.68	25.47	15.47
LASER	0.03	0.015	0.59	0.03	0.025	0.0049
LW	0.028	2.98	1.90	0.03	0.011	0.011
MORTGAGE	9.24	8.23	2.41	1.45	0.30	0.023
PL	0.117	0.29	0.29	2.118	0.047	0.029
PLASTIC	11.71	20.32	2.791	8.62	4.20	2.17
QUAKE	0.07	0.42	0.04	0.07	0.96	0.036
SN	0.026	0.40	2.95	0.027	0.026	0.024
STOCK	180.89	302.43	3.88	12.23	8.92	4.69
TREASURY	11.16	9.91	2.93	2.02	0.43	0.068
AVERAGE	**22.46**	**30.29**	**9.31**	**10.02**	**6.29**	**4.28**

In Figure 6, the comparison of the proposed machine learning model (PROPOSED) with other models on classification datasets was conducted using a script in the R programming language, which evaluated the p-value to assess the statistical significance of performance differences. The p-value serves as a measure of the likelihood that observed results are due to random variation rather than genuine differences between the models. p-values below the commonly accepted threshold of 0.05 indicate that the differences are statistically significant and not due to random chance. The comparison between PROPOSED and the ADAM model yielded a p-value of 1.9e-07, which is extremely low. This indicates that the performance difference between the two models is highly significant, rejecting the possibility that the superior performance of PROPOSED is due to random variability. This result highlights the clear advantage of PROPOSED over ADAM. In the case of PROPOSED versus BFGS, the p-value was 1.1e-08, even smaller than the comparison with ADAM. This extremely small p-value further underscores that PROPOSED significantly outperforms BFGS, demonstrating that the performance gap is exceptionally strong and statistically meaningful. For the comparison of PROPOSED with GENETIC, the p-value was 6.9e-08. Although this value is slightly higher than the previous ones, it is still extraordinarily low, indicating that PROPOSED's performance is clearly superior. This reinforces the consistency of PROPOSED when compared to models relying on genetic algorithms. The p-value for the comparison of PROPOSED with RBF was 1.9e-07, similar in magnitude to the comparison with ADAM. This consistency indicates that the superiority of PROPOSED over RBF is statistically significant, showcasing the proposed model's robust performance against a model with a different architecture. Finally, the comparison of PROPOSED with NNC produced a p-value of 4.3e-08. This very low value demonstrates the strong superiority of PROPOSED over a neural network classifier (NNC), further emphasizing the general observation that the proposed model performs better than the others, regardless of the specific technique employed by each competing model. Overall, the exceptionally low p-values for all comparisons provide evidence of the statistically significant superiority of PROPOSED compared to the other models. The results highlight that the performance of

the proposed model is not only better but also consistently superior, with differences that are robust and highly reliable from a statistical perspective.

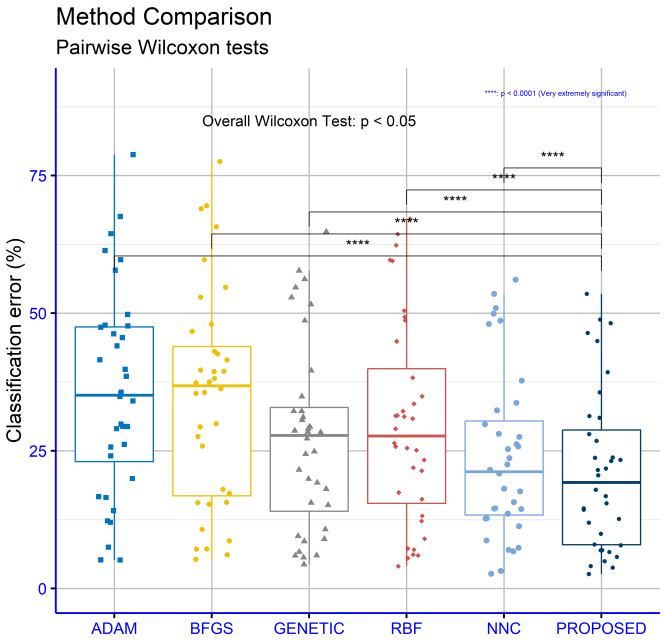

Figure 6. Statistical comparison of the machine learning models for the classification datasets.

The comparison of the proposed machine learning model (PROPOSED) with other models Figure 7 on regression datasets was conducted using statistical analysis to determine the p-values, which measure the significance of performance differences between the models. A low p-value indicates that the observed differences are statistically significant and unlikely to have occurred by chance. The results reveal a consistent and robust superiority of PROPOSED over the other models. When comparing PROPOSED with ADAM, the p-value was 0.00013. This low value indicates a statistically significant difference in performance, confirming that the observed superiority of PROPOSED is unlikely to be due to random variation. In the case of BFGS, the p-value was even lower at 9.5e-07. This extremely small value strongly supports the conclusion that PROPOSED significantly outperforms BFGS, underscoring its robustness and reliability. The comparison of PROPOSED with GENETIC yielded a p-value of 0.0001. This result reaffirms the statistical significance of the performance gap between the two models, highlighting PROPOSED's consistent advantage over GENETIC. For NEAT, the p-value was 0.0014, which, while slightly higher than the previous values, is still well below the threshold of 0.05. This demonstrates a statistically significant difference in favor of PROPOSED, confirming its superior performance even when compared to this specific algorithm. The comparison with RBF resulted in a p-value of 0.00042. This low value further solidifies the observation that PROPOSED consistently performs better than RBF, providing evidence of its strong adaptability and reliability across various regression datasets. Finally, the comparison with PRUNE yielded a p-value of 0.00011, which, like the others, is extremely low. This demonstrates that the performance differences between PROPOSED and PRUNE are statistically significant, further validating the superiority of the proposed model. Overall, the low p-values across all comparisons provide compelling evidence of the statistical significance of the performance advantages of PROPOSED over the other models in regression tasks. These results emphasize the consistency and reliability of the proposed model in delivering superior results across a wide range of regression datasets.

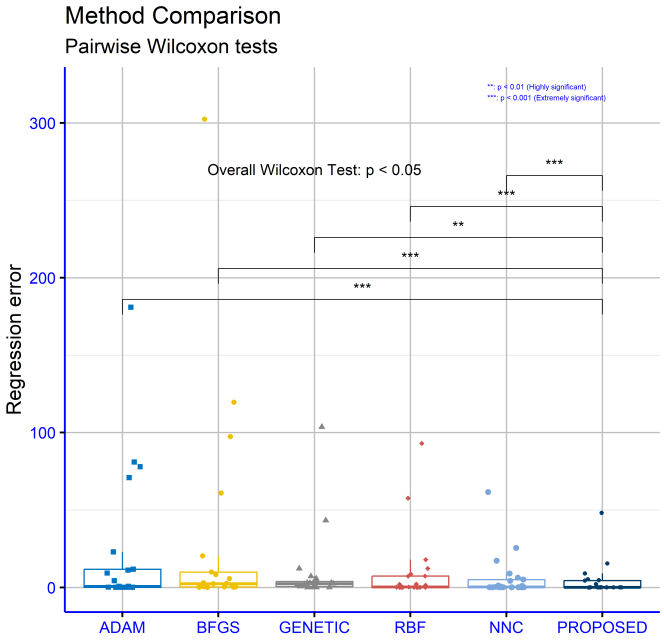

Figure 7. Statistical comparison between the used methods for the regression datasets.

Another experiment was conducted where the parameter N_K was altered in the range $[5, \ldots, 50]$ and the results for the regression datasets are depicted in Figure 8.

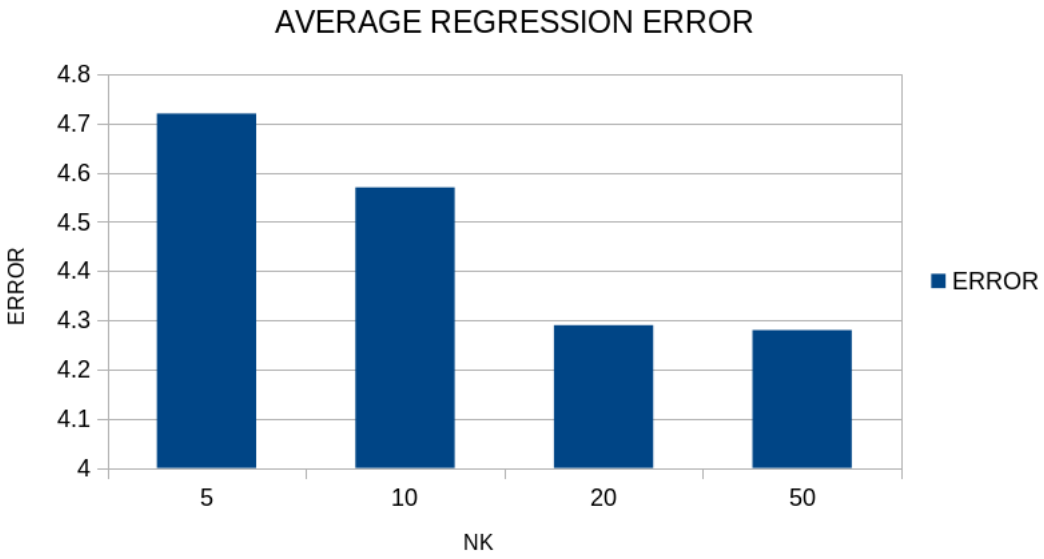

Figure 8. Average regression error for the regression datasets and the proposed method using a variety of values for the parameter N_K.

Additionally a series of experiments was conducted where the parameter N_I was changed from 10 to 40 and the results are graphically presented in Figure 9.

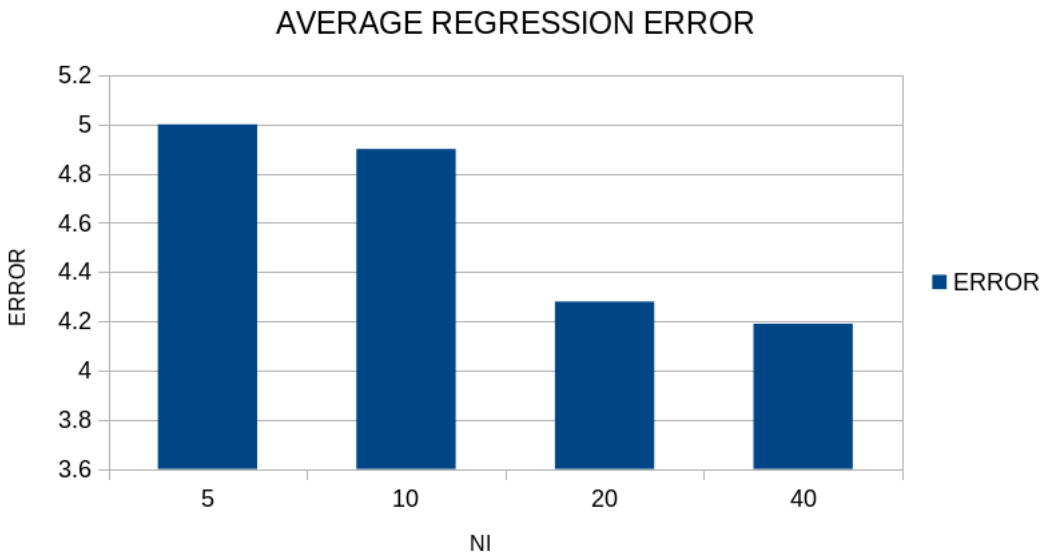

Figure 9. Experimental results for the regression datasets and the proposed method using a variety of values for the parameter N_I.

Experimental measurements show that as the parameter N_I increases, the regression error decreases, indicating that higher values of N_I contribute to improved model performance in regression tasks. Similarly, increasing the parameter N_K also results in a reduction in regression error. This relationship highlights the importance of N_I and N_K in optimizing the model's accuracy, possibly by influencing the complexity or capacity of the neural network. These findings underscore the significance of tuning N_I and N_K to achieve optimal performance in regression applications.

4. Conclusions

The article presents a method for constructing artificial neural networks by integrating Grammatical Evolution (GE) with a modified Genetic Algorithm (GA) to improve generalization properties and reduce overfitting. The method proves to be effective in designing neural network architectures and optimizing their parameters. The modified genetic algorithm avoids local minima during training and addresses overfitting by applying penalty factors to the fitness function, ensuring better generalization to unseen data. The method was evaluated on a variety of classification and regression datasets from diverse fields, including physics, chemistry, medicine, and economics. Comparative results indicate that the proposed method achieves lower error rates on average compared to traditional optimization and machine learning techniques, highlighting its stability and adaptability. The results, analyzed through statistical metrics such as p-values, provide strong evidence of the method's superiority over competing models in both classification and regression tasks. A key innovation of the method is the combination of dynamic architecture generation and parameter optimization within a unified framework. This approach not only enhances performance but also reduces the computational complexity associated with manually designing neural networks. Additionally, the use of constraint techniques in the genetic algorithm ensures the preservation of the neural network structure while enabling controlled optimization of parameters. Future explorations could focus on testing the method on larger and more complex datasets, such as those encountered in image recognition, natural language processing, and genomics, to evaluate its scalability and effectiveness in real-world applications. Furthermore, the integration of other global optimization methods, such as Particle Swarm Optimization, Simulated Annealing, or Differential Evolution, could be considered to further enhance the algorithm's robustness and convergence speed. Concurrently, the inclusion of regularization techniques, such as

dropout or batch normalization, could improve the method's generalization capabilities even further. Reducing computational cost is another important area of investigation, and the method could be adapted to leverage parallel computing architectures, such as GPUs or distributed systems, making it feasible for training on large datasets or for real-time applications. Finally, customizing the grammar used in Grammatical Evolution based on the specific characteristics of individual fields could improve the method's performance in specialized tasks, such as time-series forecasting or anomaly detection in cybersecurity.

Author Contributions: V.C. and I.G.T. conducted the experiments, employing several datasets and provided the comparative experiments. D.T. and V.C. performed the statistical analysis and prepared the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Institutional Review Board Statement: Not applicable.

Institutional Review Board Statement: Not applicable.

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

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

References

1. 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. M. Egmont-Petersen, D. de Ridder, H. Handels, Image processing with neural networks—a review, Pattern Recognition **35**, pp. 2279-2301, 2002.
4. G.Peter Zhang, Time series forecasting using a hybrid ARIMA and neural network model, Neurocomputing **50**, pp. 159-175, 2003.
5. 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.
6. P. Baldi, K. Cranmer, T. Faucett et al, Parameterized neural networks for high-energy physics, Eur. Phys. J. C **76**, 2016.
7. 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.
8. Vora, K., & Yagnik, S. (2014). A survey on backpropagation algorithms for feedforward neural networks. International Journal of Engineering Development and Research, 1(3), 193-197.
9. 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.
10. 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.
11. Hermanto, R. P. S., & Nugroho, A. (2018). Waiting-time estimation in bank customer queues using RPROP neural networks. Procedia Computer Science, 135, 35-42.
12. 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.
13. 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.
14. 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.
15. 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.
16. 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.
17. 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.

18. 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.
19. 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.
20. 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.
21. K-Su Oh, K. Jung, GPU implementation of neural networks, Pattern Recognition **37**, pp. 1311-1314, 2004.
22. M. Zhang, K. Hibi, J. Inoue, GPU-accelerated artificial neural network potential for molecular dynamics simulation, Computer Physics Communications **285**, 108655, 2023.
23. S.J. Nowlan and G.E. Hinton, Simplifying neural networks by soft weight sharing, Neural Computation 4, pp. 473-493, 1992.
24. Nowlan, S. J., & Hinton, G. E. (2018). Simplifying neural networks by soft weight sharing. In The mathematics of generalization (pp. 373-394). CRC Press.
25. 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.
26. M. Augasta and T. Kathirvalavakumar, Pruning algorithms of neural networks — a comparative study, Central European Journal of Computer Science, 2003.
27. Lutz Prechelt, Automatic early stopping using cross validation: quantifying the criteria, Neural Networks **11**, pp. 761-767, 1998.
28. 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.
29. 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.
30. 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.
31. 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.
32. P.G. Benardos, G.C. Vosniakos, Optimizing feedforward artificial neural network architecture, Engineering Applications of Artificial Intelligence **20**, pp. 365-382, 2007.
33. B.A. Garro, R.A. Vázquez, Designing Artificial Neural Networks Using Particle Swarm Optimization Algorithms, Computational Intelligence and Neuroscience, 369298, 2015.
34. M. O'Neill, C. Ryan, Grammatical evolution, IEEE Trans. Evol. Comput. **5,**pp. 349–358, 2001.
35. I.G. Tsoulos, D. Gavrilis, E. Glavas, Neural network construction and training using grammatical evolution, Neurocomputing **72**, pp. 269-277, 2008.
36. G.V. Papamokos, I.G. Tsoulos, I.N. Demetropoulos, E. Glavas, Location of amide I mode of vibration in computed data utilizing constructed neural networks, Expert Systems with Applications **36**, pp. 12210-12213, 2009.
37. I.G. Tsoulos, D. Gavrilis, E. Glavas, Solving differential equations with constructed neural networks, Neurocomputing **72**, pp. 2385-2391, 2009.
38. I.G. Tsoulos, G. Mitsi, A. Stavrakoudis, S. Papapetropoulos, Application of Machine Learning in a Parkinson's Disease Digital Biomarker Dataset Using Neural Network Construction (NNC) Methodology Discriminates Patient Motor Status, Frontiers in ICT 6, 10, 2019.
39. V. Christou, I.G. Tsoulos, V. Loupas, A.T. Tzallas, C. Gogos, P.S. Karvelis, N. Antoniadis, E. Glavas, N. Giannakeas, Performance and early drop prediction for higher education students using machine learning, Expert Systems with Applications **225**, 120079, 2023.
40. E.I. Toki, J. Pange, G. Tatsis, K. Plachouras, I.G. Tsoulos, Utilizing Constructed Neural Networks for Autism Screening, Applied Sciences **14**, 3053, 2024.
41. J. W. Backus. The Syntax and Semantics of the Proposed International Algebraic Language of the Zurich ACM-GAMM Conference. Proceedings of the International Conference on Information Processing, UNESCO, 1959, pp.125-132.
42. C. Ryan, J. Collins, M. O'Neill, Grammatical evolution: Evolving programs for an arbitrary language. In: Banzhaf, W., Poli, R., Schoenauer, M., Fogarty, T.C. (eds) Genetic Programming. EuroGP 1998. Lecture Notes in Computer Science, vol 1391. Springer, Berlin, Heidelberg, 1998.
43. M. O'Neill, M., C. Ryan, Evolving Multi-line Compilable C Programs. In: Poli, R., Nordin, P., Langdon, W.B., Fogarty, T.C. (eds) Genetic Programming. EuroGP 1999. Lecture Notes in Computer Science, vol 1598. Springer, Berlin, Heidelberg, 1999.
44. A.O. Puente, R. S. Alfonso, M. A. Moreno, Automatic composition of music by means of grammatical evolution, In: APL '02: Proceedings of the 2002 conference on APL: array processing languages: lore, problems, and applications July 2002 Pages 148–155.
45. E. Galván-López, J.M. Swafford, M. O'Neill, A. Brabazon, Evolving a Ms. PacMan Controller Using Grammatical Evolution. In: , et al. Applications of Evolutionary Computation. EvoApplications 2010. Lecture Notes in Computer Science, vol 6024. Springer, Berlin, Heidelberg, 2010.
46. N. Shaker, M. Nicolau, G. N. Yannakakis, J. Togelius, M. O'Neill, Evolving levels for Super Mario Bros using grammatical evolution, 2012 IEEE Conference on Computational Intelligence and Games (CIG), 2012, pp. 304-31.

47. D. Martínez-Rodríguez, J. M. Colmenar, J. I. Hidalgo, R.J. Villanueva Micó, S. Salcedo-Sanz, Particle swarm grammatical evolution for energy demand estimation, Energy Science and Engineering **8**, pp. 1068-1079, 2020.
48. C. Ryan, M. Kshirsagar, G. Vaidya, G. et al. Design of a cryptographically secure pseudo random number generator with grammatical evolution. Sci Rep **12**, 8602, 2022.
49. C. Martín, D. Quintana, P. Isasi, Grammatical Evolution-based ensembles for algorithmic trading, Applied Soft Computing **84**, 105713, 2019.
50. 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).
51. M. Kelly, R. Longjohn, K. Nottingham, The UCI Machine Learning Repository, https://archive.ics.uci.edu.
52. 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.
53. 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.
54. 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.
55. J.R. Quinlan, Simplifying Decision Trees. International Journal of Man-Machine Studies **27**, pp. 221-234, 1987.
56. T. Shultz, D. Mareschal, W. Schmidt, Modeling Cognitive Development on Balance Scale Phenomena, Machine Learning **16**, pp. 59-88, 1994.
57. 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.
58. R. Setiono , W.K. Leow, FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks, Applied Intelligence **12**, pp. 15-25, 2000.
59. 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.
60. 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.
61. 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.
62. I. Kononenko, E. Šimec, M. Robnik-Šikonja, Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF, Applied Intelligence **7**, pp. 39–55, 1997
63. 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.
64. J.G. Dy , C.E. Brodley, Feature Selection for Unsupervised Learning, The Journal of Machine Learning Research **5**, pp 845–889, 2004.
65. S. J. Perantonis, V. Virvilis, Input Feature Extraction for Multilayered Perceptrons Using Supervised Principal Component Analysis, Neural Processing Letters **10**, pp 243–252, 1999.
66. J. Garcke, M. Griebel, Classification with sparse grids using simplicial basis functions, Intell. Data Anal. **6**, pp. 483-502, 2002.
67. J. Mcdermott, R.S. Forsyth, Diagnosing a disorder in a classification benchmark, Pattern Recognition Letters **73**, pp. 41-43, 2016.
68. 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.
69. 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.
70. 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.
71. 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.
72. 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.
73. 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.
74. 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.
75. T. Hastie, R. Tibshirani, Non-parametric logistic and proportional odds regression, JRSS-C (Applied Statistics) **36**, pp. 260–276, 1987.

76. M. Dash, H. Liu, P. Scheuermann, K. L. Tan, Fast hierarchical clustering and its validation, Data & Knowledge Engineering **44**, pp 109–138, 2003.
77. 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.
78. 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.
79. 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.
80. 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.
81. 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.
82. P. Zhong, M. Fukushima, Regularized nonsmooth Newton method for multi-class support vector machines, Optimization Methods and Software **22**, pp. 225-236, 2007.
83. 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.
84. 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
85. M. Koivisto, K. Sood, Exact Bayesian Structure Discovery in Bayesian Networks, The Journal of Machine Learning Research **5**, pp. 549–573, 2004.
86. 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
87. 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).
88. I.Cheng Yeh, Modeling of strength of high performance concrete using artificial neural networks, Cement and Concrete Research. **28**, pp. 1797-1808, 1998.
89. Friedman, J. (1991): Multivariate Adaptative Regression Splines. Annals of Statistics, 19:1, 1--141.
90. D. Harrison and D.L. Rubinfeld, Hedonic prices and the demand for clean ai, J. Environ. Economics & Management **5**, pp. 81-102, 1978.
91. M.J.D Powell, A Tolerant Algorithm for Linearly Constrained Optimization Calculations, Mathematical Programming **45**, pp. 547-566, 1989.
92. J. Park and I. W. Sandberg, Universal Approximation Using Radial-Basis-Function Networks, Neural Computation **3**, pp. 246-257, 1991.
93. G.A. Montazer, D. Giveki, M. Karami, H. Rastegar, Radial basis function neural networks: A review. Comput. Rev. J **1**, pp. 52-74, 2018.