Article

Local Crossover: A new genetic operator for Grammatical Evolution

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: In this work, a new genetic crossover operator is proposed, which can be used to problems solved by the Grammatical Evolution technique. This new operator intensively applies the one - point crossover procedure to randomly selected chromosomes with the aim of drastically reducing their fitness value. To apply the one point crossover method, a set of randomly selected chromosomes is selected from the current population. This new operator was applied to two techniques from the recent literature that exploit Grammatical Evolution: artificial neural network construction and rule construction. In both case studies, an extensive set of classification problems and data fitting problems were incorporated to estimate the effectiveness of the proposed genetic operator. The proposed operator significantly reduced both the classification error on the classification datasets and the feature learning error on the fitting datasets, compared to other machine learning techniques and also to the original models before applying the new operator.

Keywords: Genetic algorithms; Genetic Programming ; Grammatical Evolution; Genetic operators

Citation: Tsoulos, I.G.; Charilogis, V.; Tsalikakis, D. Local Crossover: A new genetic operator for Grammatical Evolution. *Journal Not Specified* **2024**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

1. Introduction

Genetic algorithms are stochastic optimization algorithms originated in the work of Holland [1]. They belong to a wide area of optimization algorithms called evolutionary techniques [2]. Genetic algorithms initiate by formulating candidate solutions of the objective problem. These solutions are evolved through a series of processes that mimic natural evolution, such as selection, crossover and mutation [3–5]. The genetic algorithms were incorporated in a variety of problems, such as networking problems [6], problems arise in robotics [7,8], energy problems [9,10], medicine problems [11,12], agriculture problems [13] etc.

Grammatical Evolution [15] is an integer based genetic algorithm, where each chromosome represents a series of production rules derived from a Backus–Naur form (BNF) grammar [16]. Grammatical Evolution can be utilized to produce programs in any programming language. This method was used in a variety of cases derived from real - world problems, such as data fitting [17,18], credit classification [19], detection of network attacks [20], solving differential equations [21], monitoring the quality of drinking water [22], construction of optimization methods [23], application in trigonometric problems [24], composition of music [25], constructing neural networks [26,27], production of numeric constants with a variable number of digits [28], video games [29,30], energy problems [31], combinatorial optimization [32], security topics [33], automatic construction of decision trees [34], circuit design [35], discovering taxonomies in Wikipedia [36], trading algorithms [37], bioinformatics [38], modeling glycemia in humans [39], etc.

Grammatical Evolution has been extended by many researches in recent bibliography. Among these extensions there are works, such as the Weighted Hierarchical Grammatical Evolution [40], which proposed a novel technique to map genotypes to phenotypes. Also, the method of Structured Grammatical Evolution [41,42], proposed a one-to-one mapping for chromosomes and the non - terminal symbols of the grammar. Also, O'Neill et al. suggested the usage of shape grammars in the Grammatical Evolution for evolutionary design [43]. Also the πGrammatical Evolution method [44] was suggested as a modification of the Grammatical Evolution, where a position - independent mapping was proposed. Another interesting work was the incorporation of the Particle Swarm Optimization(PSO) [45] to create expressions in Grammatical Evolution. This method was named Grammatical Swarm [46,47] in the relevant literature. Moreover, the method of Probabilistic Grammatical Evolution [48] has been introduced recently, where a stochastic procedure was used as a mapping mechanism. Recently, the optimization method of Fireworks algorithm [49] was applied as a learning algorithm for the Grammatical Evolution procedure [50]. Contreras et al. suggested the combination of Grammatical Evolution and some ideas from interval analysis, to solve problems with uncertainty [51]. Grammatical evolution has also been extended using programming techniques [52,53] or Christiansen grammars [54].

Additionally, many researchers have developed and published open source software for Grammatical Evolution, such as the graphical user interface (GUI) application of Grammatical Evolution in Java (GEVA)[55], a Java implementation called jGE [56], an R implementation of Grammatical Evolution called gramEvol (Grammatical Evolution for R) [57], the GRAPE software that implemented Grammatical Evolution in Python [58], the GeLab [59] software that suggested a Matlab toolbox for Grammatical Evolution, a software which produced classification programs with Grammatical Evolution called GenClass [60], the QFc software [61] the used to create new features from the initial features with the assistance of Grammatical Evolution etc.

In this work, a new genetic operator for Grammatical Evolution is introduced, which is based on the one crossover technique. The new genetic operator is stochastically applied to the genetic population, randomly selecting a set of chromosomes on which to apply it. For each randomly selected chromosome, a group of chromosomes is stochastically formed from the current genetic population. Afterwards, a one - point crossover operation is performed between the selected chromosome and each of the generated group to search for a lower value of the fitness function. The new genetic operator was applied in two distinct cases of Grammatical Evolution methods: in the rule construction technique introduced recently [62] and in the neural network construction technique [63]. These machine learning tools were applied on a wide series of classification and regression datasets from the relevant literature and the experimental results indicated a reduction in classification or regression error from the application of the new genetic operator.

The main components of the proposed technique are the following:

1. The method can be applied as a genetic operator to all problems solved by the Grammatical Evolution technique and the only information it exploits is the fitness function of the problem.
2. The method has no dependence on the grammar of the objective problem.
3. By using an application rate the user can require fewer or more applications of the new operator.
4. Although this operator requires significant computing time for its execution, its application between chromosomes can be done using parallel techniques, since there is no dependency between its successive applications.
5. Simple linear operations are required for its implementation, such as crossing a point between chromosomes.
6. The new genetic operator could theoretically be applied to other forms of genetic algorithms beyond Grammatical Evolution.

The following sections have this structure: in section 2 the basic principles of Grammatical Evolution are discussed as well as the current genetic operator. In section 3 the used datasets are provided accompanied by the experimental results and in section 4 some conclusions are discussed in detail.

2. Materials and Methods

2.1. The Grammatical Evolution method

The Grammatical Evolution considers chromosomes as set of production rules in the provided BNF grammar. BNF grammar is usually denoted as a set $G = (N, T, S, P)$, with the following assumptions:

- N defines the set of non-terminal symbols.
- T is a set that contains the terminal symbols.
- S is a non - terminal symbol that corresponds to the that start symbol of the grammar.
- P is the set of production rules. These rules are in the form $A \to a$ or $A \to aB,\ A, B \in N,\ a \in T$. For Grammatical Evolution a sequence number is assigned to every production rule.

The grammar used for the rule machine learning method is outlined in Figure 1 and the grammar used for the neural network construction technique is displayed in figure 2.

Figure 1. The BNF grammar for the rule construction method.

```
<S>::= <ifexpr> value=<expr> else value=<expr>  (0)
<ifexpr>::= if(<boolexpr>) value=<expr> (0)
            |<ifexpr> else if(<boolexpr>) value=<expr> (1)
<boolexpr>::=<expr> <relop> <expr> (0)
            |<boolexpr> <boolop> <boolexpr> (1)
<relop>::= > (0)
           |>= (1)
           |<  (2)
           |<= (3)
           |=  (4)
           |!= (5)
<boolop>::=  & (0)
           | | (1)
<expr> ::=  (<expr> <op> <expr>)  (0)
           | <func> ( <expr> )    (1)
           |<terminal>            (2)
<op> ::=     +      (0)
           | -      (1)
           | *      (2)
           | /      (3)
<func> ::=   sin  (0)
           | cos  (1)
           |exp   (2)
           |log   (3)
<terminal>::=<xlist>                (0)
           |<digitlist>.<digitlist> (1)
<xlist>::=x1    (0)
           | x2 (1)
           .........
           | xD (D)
<digitlist>::=<digit>                  (0)
           | <digit><digit>            (1)
           | <digit><digit><digit>     (2)
<digit>  ::= 0 (0)| 1 (1)
           | 2 (2)| 3 (3)
           | 4 (4)| 5 (5)
           | 6 (6)| 7 (7)
           | 8 (8)| 9 (9)
```

Figure 2. The grammar for the neural network construction method.

```
S:=<sigexpr>                          (0)
<sigexpr>::=<Node>                    (0)
           | <Node> + <sigexpr>       (1)
<Node>::=<number>*sig(<sum>+<number>) (0)
<sum>::= <number>*<xxlist>            (0)
           |    <sum>+<sum>           (1)
<xxlist>::= x1        (0)
             |    x2  (1)
             ..............
             |    xD  (D-1)
<number>::= (<digitlist>.<digitlist>)        (0)
             |    (-<digitlist>.<digitlist>) (1)
<digitlist>::= <digit>            (0)
             | <digit><digitlist> (1)
<digit>::= 0      (0)
             |  1 (1)
             ...........
             |  9 (9)
```

The notation < > is used to enclose Non-terminal symbols. The numbers at the end of the production rules represent the sequence number of each rule. The parameter d stands for the dimension of the used dataset (number of features). Grammatical Evolution produces valid expressions by starting from the starting symbol S and by following the production rules. The method selects the following production rules according to the scheme:

- **Get** the next element V from the current chromosome.
- **Select** the next rule as:
$$\text{Rule} = V \bmod \text{NR} \tag{1}$$
where the value NR represents the total number of production rules for the under processing non – terminal symbol.

For a dataset with three features (x_1, x_2, x_3) an example of rule construction method could be the following:

$$\text{if}(x_1 > 2 + \sin(x_3))\text{value} = 1 + \cos(x_2) \text{ else value} = 1 + x_1$$

The terminal symbol value is used to denote the final outcome of the rule method. For the same dataset an example of neural network constructed by the Grammatical Evolution could be the following:

$$\text{NNC}(x) = 2.45\text{sig}(1.9x_1 + 3.11x_3 + 2.5) + 5.9\text{sig}(10.8x_2 + 6.25)$$

This grammar of Figure 2 is able to construct artificial neural networks in the form:

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

The vector $\overrightarrow{x}$ represents the input pattern, while the vector $\overrightarrow{p}$ stands for the parameter set of the neural network. The parameter H defines the number of weights or processing units. These networks have one processing level, but the grammar could be easily extended to produce neural networks of additional levels. The function $\mathrm{sig}(x)$ used in the previous example denotes the sigmoid function given by:

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

used in the majority of cases of neural networks. Of course, the Grammatical Evolution procedure can construct neural networks with different activation functions or neural networks that use a mix of activation functions.

2.2. The genetic algorithm

The two machine learning techniques, in which the new genetic operator was applied, follow a series of similar execution steps, which are presented in detail below:

1. **Initialization step**.
   (a) **Set** $k = 0$ the generation counter.
   (b) **Set** N_g the maximum number of allowed generations.
   (c) **Set** as N_c the total number of chromosomes in the genetic population.
   (d) **Set** as p_s the selection rate, where $p_s \leq 1$.
   (e) **Set** as p_m the mutation rate, where $p_m \leq 1$
   (f) **Set** as p_{cr} the rate for the application of the new crossover operator, where $p_{cr} \leq 1$
   (g) **Set** as N_{cr} the amount of chromosomes the will be selected for each chromosome where the new crossover operator will be applied.
2. **Fitness Calculation step**.
   (a) **For** $i = 1, \ldots, N_c$ **do**
      i. **Set** as f_i the fitness of chromosome i. For the case of rule creation model the grammar of Figure 1 is applied while for the case of neural network construction the grammar of Figure 2 is utilized.
   (b) **EndFor**
3. **Application of genetic operations step.**
   (a) **Apply** the selection procedure: In the first phase, the chromosomes are sorted according to their fitness. The $(1 - p_s) \times N_c$ best of these are transmitted unchanged to the next generation, while the rest will be replaced by chromosomes produced through crossover and mutation.
   (b) **Apply** the crossover procedure: During this procedure $p_s \times N_c$ offsprings are produced from the current population. For every set $(\tilde{z}, \tilde{w})$ of produced children, two distinct chromosomes (z, w) are selected from the population with tournament selection. The offsprings $(\tilde{z}, \tilde{w})$ are formulated using the one - point crossover procedure, graphically outlined in Figure 3.
   (c) **Perform** the mutation procedure. A random number $r \in [0, 1]$ is selected for each element of every chromosome and this element is changed randomly if $r \leq p_m$.
   (d) **Apply** the new crossover operator: For each chromosome $g_i,\ i = 1, \ldots, N_c$ a random number $r \in [0, 1]$ is selected. If $r \leq p_{cr}$ then execute the procedure described in subsection 2.3 on g_i.
4. **Check for termination step**.
   (a) **Set** $k = k + 1$
   (b) **If** $k \leq N_g$ goto Fitness Calculation Step, **else** terminate.

The previous algorithm is also graphically illustrated in the flowchart of Figure 4.

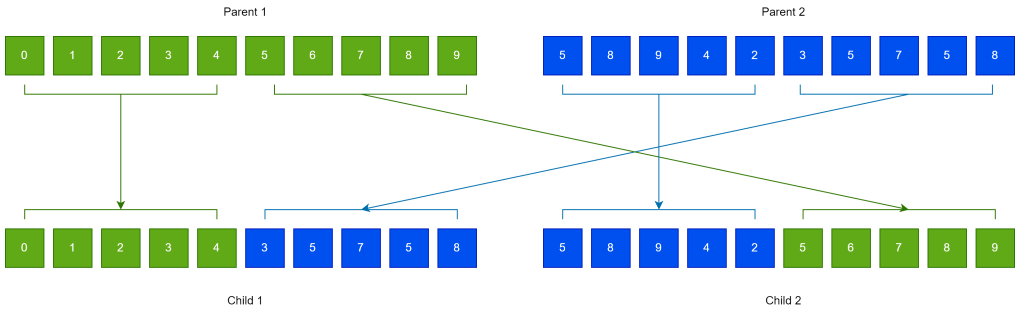

Figure 3. An example of the method of one - point crossover. This method is used as the crossover procedure in the Grammatical Evolution procedure.

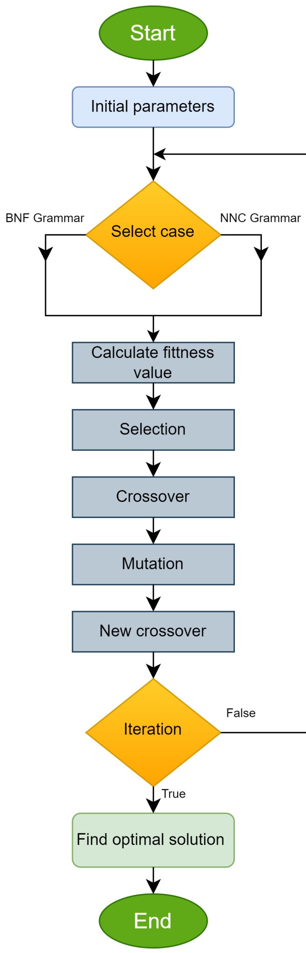

Figure 4. A flowchart indicating the main steps of the used genetic algorithm.

2.3. *The new crossover operator*

The new crossover operator executes the one - point crossover method on a selected chromosome using a set of randomly chosen chromosomes from the current population. The steps of this procedure are listed below:

1. **Set** as g the chromosome where the operator will be applied and as f_g the corresponding fitness value.
2. **Create** the set $C = \{x_1, x_2, \ldots, x_{N_{cr}}\}$ of N_{cr} randomly selected chromosomes.
3. **For** $i = 1, \ldots, N_{cr}$ **do**
   (a) **Perform** one - point crossover between g and x_i. This procedure produces the offsprings g_1 and g_2 with associated fitness values f_{g_1} and f_{g_2}.

(b) **If** $f_{g_1} \leq f_g$ **then**
  i. $g = g_1$
(c) **else if** $f_{g_2} \leq f_g$ **then**
  i. $g = g_2$
(d) **Endif**

4. **EndFor**

3. Results

The suggested genetic operator was tested on a set of classification and regression datasets obtained from the recent bibliography and relevant websites. The internet sources for these datasets are the following websites:

1. The UCI dataset repository, https://archive.ics.uci.edu/ml/index.php(accessed on 25 September 2024)[64]
2. The Keel repository, https://sci2s.ugr.es/keel/datasets.php(accessed on 25 September 2024)[65].
3. The Statlib URL http://lib.stat.cmu.edu/datasets/(accessed on 25 September 2024).

3.1. Classification datasets

A series of classification datasets was used in the conducted experiments:

1. **Appendictis** a medical dataset, proposed in [66].
2. **Australian** dataset [67], suggested for credit card transactions.
3. **Balance** dataset [68], used in a series of psychological experiments.
4. **Circular** dataset, that is an artificial dataset.
5. **Cleveland** dataset, a medical dataset[69,70].
6. **Dermatology** dataset [71], a dataset related to dermatological deceases.
7. **Ecoli** dataset, a dataset related to protein problems[72].
8. **Fert** dataset, which is used to detect any relation of fertility and sperm concentration.
9. **Haberman** dataset, which is related to breast cancer.
10. **Hayes roth** dataset, a datatet provided by [73].
11. **Heart** dataset [74], a medical dataset for the prediction of heart diseases.
12. **HeartAttack** dataset, a medical dataset related to heart diseases.
13. **HouseVotes** dataset [75], related to votes in the U.S. House of Representatives.
14. **Glass** dataset, which contains glass component analysis for glass pieces that belong to 6 classes.
15. **Liverdisorder** dataset [76], a medical dataset for the detection of liver disorders.
16. **Mammographic** dataset [78], a medical dataset about breast cancer.
17. **Parkinsons** dataset, a dataset related to the detection of Parkinson's disease (PD)[77].
18. **Pima** dataset [79], a medical dataset used in the detection of diabetes.
19. **Popfailures** dataset [80], that to do with climate related measurements.
20. **Regions2** dataset, a dataset used in the detection of hepatitis C [81].
21. **Saheart** dataset [82], involved to the detection of heart diseases.
22. **Segment** dataset [83], which contains information regarding image processing.
23. **Spiral** dataset, which is an artificial dataset.
24. **Student** dataset [84], a dataset for measurements in schools.
25. **Wdbc** dataset [85], used in cancer detection.
26. **Wine** dataset, which contains information about wines. [86,87].
27. **Eeg** datasets, a medical dataset contains EEG measurements [88]. From this dataset the next cases were used: Z_F_S, Z_O_N_F_S, ZO_NF_S and ZONF_S.
28. **Zoo** dataset [89], used to estimate the category of some animals.

3.2. Regression datasets

A series of regression datasets was utilized in the conducted experiments:

1. **Abalone** dataset [90], a dataset used to predict the age of abalones.
2. **Airfoil** dataset, provided by NASA [91].
3. **BK** dataset [92], related to the prediction of points in a basketball game.
4. **BL** dataset, it contains measurements from electricity experiments.
5. **Baseball** dataset, used to estimate the income of baseball players.
6. **Concrete** dataset [93], is a dataset related to the durability of cements in public works.
7. **Dee** dataset, that contains measurements regarding the price of electricity.
8. **FY,** a dataset that contains measurements for the longevity of fruit flies.
9. **HO** dataset, provided from the the STALIB repository.
10. **Housing** dataset, presented in [94].
11. **Laser** dataset, that contains measurements from laser experiments.
12. **LW** dataset, that contains measurements regarding the weight of babies.
13. **MORTGAGE** dataset, that contains economic data from USA.
14. **MUNDIAL,** downloaded from the STALIB repository.
15. **PL** dataset, downloaded from the STALIB repository.
16. **QUAKE** dataset, that contains measurements from earthquakes.
17. **REALESTATE,** downloaded from the STALIB repository.
18. **SN** dataset, that is used in an experiment related to trellising and pruning.
19. **Treasury** dataset, which is a dataset regarding the economy of USA.
20. **TZ** dataset, founded in the STALIB repository.
21. **VE** dataset, originated in the STALIB repository.

3.3. Experimental results

The code used in the experiments was coded in ANSI C++ using the Optimus optimization environment, available from https://github.com/itsoulos/GlobalOptimus/(accessed on 25 September 2024). All the experiments were executed 30 times and the random number generator was initialized using different seed in every execution. For the case of classification problems, the average classification error as measured on the test set was recorded and, for the case of regression datasets, the average regression error as measured on the test set was recorded. In order to validate the experimental results, the well -known method of ten - fold cross validation was incorporated. Table 1 depicts the experimental settings for the conducted experiments.

Table 1. All the value for the used experimental parameters.

PARAMETER	MEANING	VALUE
N_c	Chromosomes used	500
N_g	Maximum number of generations	200
p_s	Crossover rate	0.10
p_m	Mutation rate	0.05
p_{cr}	New crossover rate	0.05
N_{cr}	New crossover items	100
H	Weights for the neural network	10

The experimental results obtained for the case of classification are depicted in Table 2 and the experimental results for the case of regression datasets are displayed in Table 3. The following notations are used in the experimental tables:

1. The column BFGS denotes the application of the BFGS method [95] to train an artificial neural network with H hidden nodes.
2. The column GEN denotes the application of a genetic algorithm [96] to train an artificial neural network with H hidden nodes. The values for the critical parameters of the genetic algorithm are included in Table **??**.
3. The column RULE refers to the simple rule construction method [14], without the incorporation of the new crossover operator.

4. The column NNC refers to the method of neural network construction [63], without the application of the new crossover operator.
5. The column RULE_CROSS represents the application of the new crossover operator the rule construction machine learning model.
6. The column NNC_CROSS depicts the results for the neural construction method with the assistance of the new genetic operator.
7. The last row under the name AVERAGE, represents the average classification or regression error for all datasets.

Table 2. Experimental results for the classification datasets. The numbers in cells stand for average classification error as calculated on the corresponding test set.

DATASET	BFGS	GEN	RULE	NNC	RULE_CROSS	NNC_CROSS
APPENDICITIS	18.00%	24.40%	14.70%	13.70%	14.80%	14.40%
AUSTRALIAN	38.13%	36.64%	14.27%	14.51%	14.46%	14.71%
BALANCE	8.64%	8.36%	28.79%	22.11%	17.47%	14.32%
CIRCULAR	6.08%	5.13%	13.25%	13.64%	9.12%	7.49%
CLEVELAND	77.55%	57.21%	48.24%	50.10%	47.52%	49.21%
DERMATOLOGY	52.92%	16.60%	43.77%	25.06%	38.00%	12.92%
ECOLI	69.52%	54.67%	55.18%	47.82%	53.48%	49.15%
FERT	23.20%	28.50%	17.40%	19.00%	17.50%	19.20%
HABERMAN	33.10%	27.80%	27.03%	28.03%	26.53%	28.37%
HAYES-ROTH	56.54%	35.85%	39.39%	35.93%	38.08%	24.08%
HEART	39.44%	26.41%	20.30%	15.78%	19.41%	15.33%
HEARTATTACK	46.67%	29.03%	23.63%	19.33%	23.70%	18.73%
HOUSEVOTES	7.13%	7.00%	3.48%	3.65%	4.51%	3.22%
GLASS	69.95%	55.09%	58.10%	57.10%	54.81%	53.82%
IONOSPHERE	13.37%	18.03%	15.06%	11.12%	14.14%	9.25%
LIVERDISORDER	42.59%	37.09%	37.09%	33.71%	35.68%	31.24%
MAMMOGRAPHIC	29.54%	16.33%	19.00%	17.78%	18.10%	17.12%
PARKINSONS	27.58%	16.58%	13.47%	12.21%	13.37%	11.47%
PIMA	35.59%	34.21%	27.85%	27.99%	27.30%	25.95%
POPFAILURES	5.24%	4.17%	5.44%	6.74%	5.02%	6.41%
REGIONS2	36.28%	33.53%	29.13%	25.52%	29.26%	24.46%
SAHEART	37.48%	34.85%	30.20%	30.52%	31.00%	28.64%
SEGMENT	68.97%	46.30%	71.51%	54.99%	61.99%	35.82%
SPIRAL	47.99%	47.67%	50.06%	48.39%	49.08%	48.04%
STUDENT	4.90%	6.75%	11.08%	5.78%	7.23%	5.06%
TRANSFUSION	25.59%	24.01%	25.19%	25.34%	24.46%	24.44%
WDBC	29.91%	7.87%	7.66%	6.95%	6.43%	6.48%
WINE	59.71%	22.88%	15.35%	14.35%	12.47%	9.88%
Z_F_S	39.37%	24.60%	16.40%	14.17%	8.77%	10.23%
Z_O_N_F_S	79.04%	64.26%	53.64%	49.18%	44.60%	42.30%
ZO_NF_S	43.04%	21.54%	14.10%	14.14%	8.39%	9.12%
ZONF_S	15.62%	4.36%	2.76%	3.14%	2.06%	2.70%
ZOO	12.10%	10.20%	14.80%	9.20%	11.10%	5.70%
AVERAGE	**36.39%**	**26.91%**	**26.28%**	**23.54%**	**23.93%**	**20.58%**

Table 3. Experimental results for the regression datasets. Numbers in table represent the average regression error as measured on the corresponding test set.

DATASET	BFGS	GEN	RULE	NNC	RULE_CROSS	NNC_CROSS
ABALONE	6.38	7.17	7.36	5.05	5.32	4.63
AIRFOIL	0.003	0.001	0.003	0.003	0.002	0.002
BK	0.36	0.26	0.02	2.32	0.037	0.15
BL	1.09	2.23	2.53	0.021	0.023	0.40
BASEBALL	119.63	64.60	65.64	59.85	61.35	58.75
CONCRETE	0.023	0.001	0.013	0.008	0.009	0.005
DEE	2.36	0.47	0.43	0.26	0.32	0.23
FY	0.19	0.65	0.041	0.058	0.046	0.049
HO	0.62	0.37	0.019	0.017	0.019	0.014
HOUSING	97.38	35.97	47.99	26.35	26.74	19.10
LASER	0.03	0.084	0.055	0.024	0.032	0.019
LW	0.26	0.54	0.012	0.011	0.013	0.017
MORTGAGE	8.23	0.40	0.20	0.30	0.13	0.21
MUNDIAL	0.05	1.22	0.038	4.47	0.049	0.76
PL	0.11	0.03	0.056	0.045	0.035	0.036
QUAKE	0.09	0.12	1.13	0.045	0.73	0.046
REALESTATE	128.94	81.19	104.74	76.78	92.49	69.77
SN	0.16	0.20	0.025	0.026	0.026	0.024
TREASURY	9.91	0.44	0.15	0.47	0.12	0.30
TZ	0.21	0.097	0.036	5.04	0.035	0.061
VE	1.92	2.43	0.028	6.61	0.043	0.084
AVERAGE	**17.99**	**9.45**	**10.98**	**8.94**	**8.93**	**7.35**

The experimental results indicate that there is a significant reduction in the mean error using the new genetic operator in both machine learning models. In a wide range of data sets, the proposed technique drastically reduces the error of data classification or fitting, as it is also represented in the graphs 5 and 6. These graphs show the number of datasets in which the application of the new genetic operator resulted in a drastic reduction in the corresponding error.

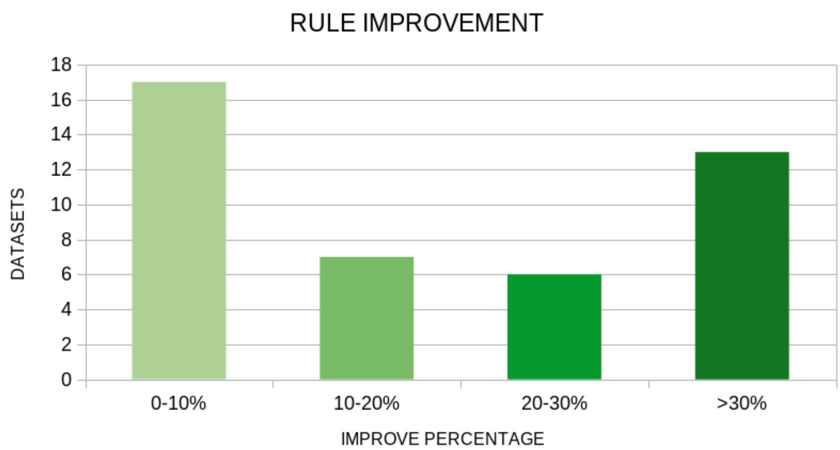

Figure 5. Number of datasets that improved in the RULE machine learning model using the proposed method. The vertical axis represents the number of data sets and the horizontal axis the percentage of reduction in error.

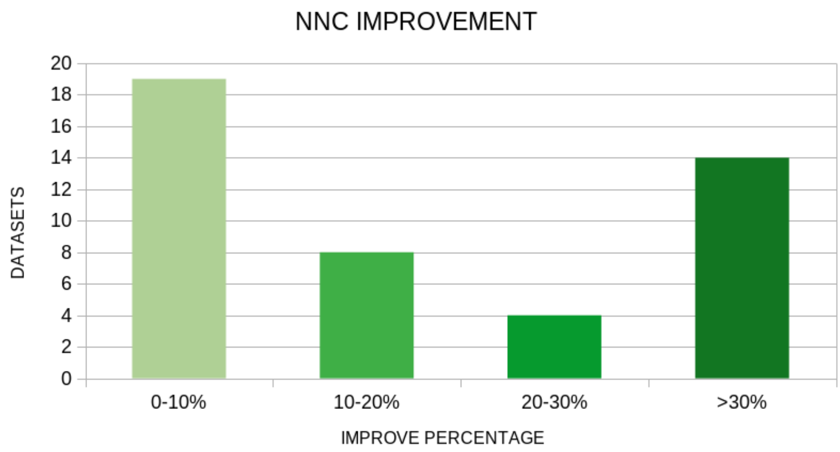

Figure 6. Number of datasets that improved in the NNC machine learning model using the proposed method. The vertical axis represents the number of data sets and the horizontal axis the percentage of reduction in error.

Furthermore, the box plots for the classification cases are shown in Figures 7 and 8 for the rule construction model and the network construction model respectively.

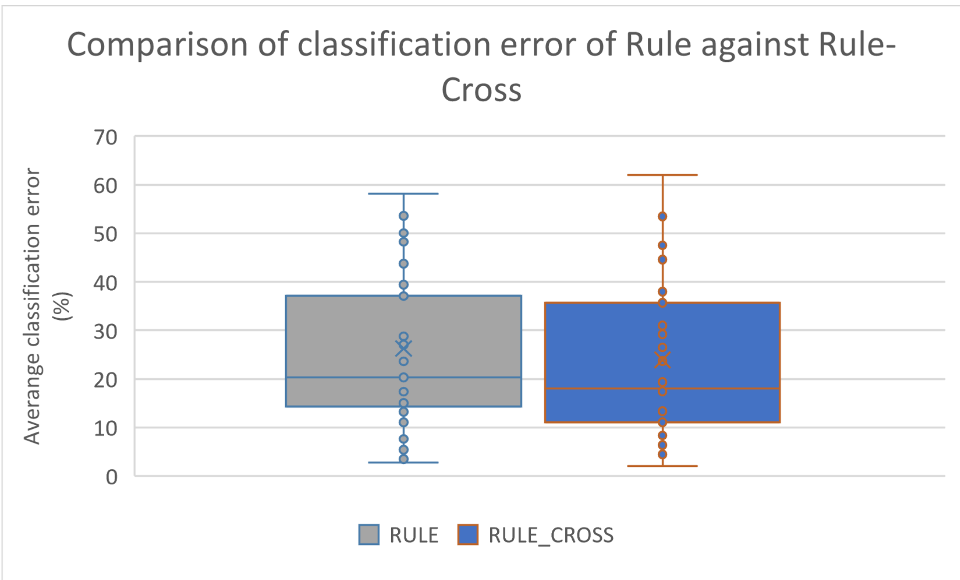

Figure 7. Box plot for the comparison between the original rule construction model and the improved one that utilizes the new crossover operator.

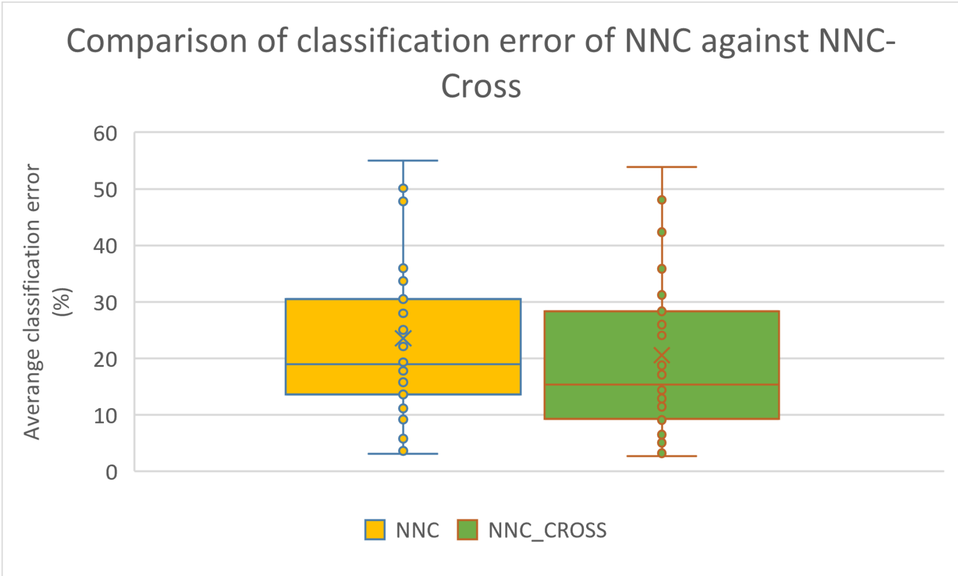

Figure 8. Box plot for the comparison between the original rule construction model and the improved one that utilizes the new crossover operator.

Box plots for the same comparisons as deduced from the results on the regression datasets are depicted in Figures 9 and 10 respectively.

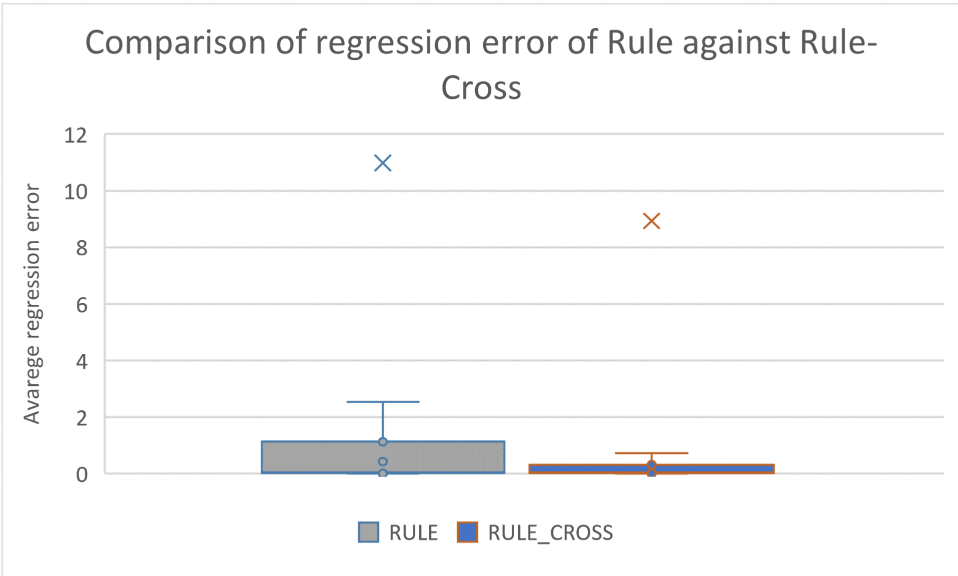

Figure 9. Box plot for the comparison between the original rule construction model and the improved one that utilizes the new crossover operator for the regression datasets.

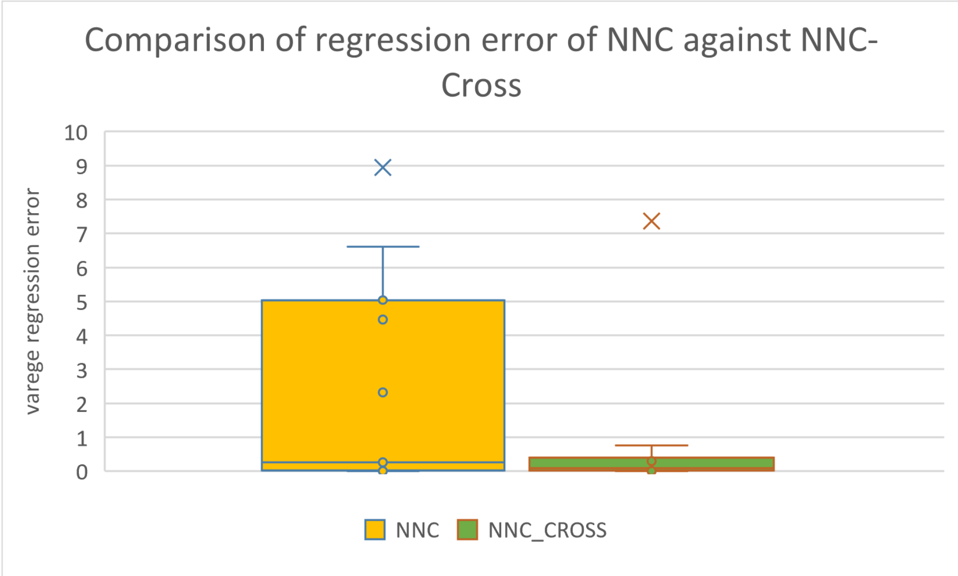

Figure 10. Box plot for the comparison between the original rule construction model and the improved one that utilizes the new crossover operator for the regression datasets.

These figures confirm the significant improvement brought about by the use of the new operator in the effectiveness of the two techniques that utilize Grammatical Evolution. This improvement appears to be greater in the datasets used in data fitting.

Also, a statistical comparison was performed between the two machine learning methods and the enhanced ones that use the new crossover operator. This comparison was performed for the classification datasets, and it is graphically outlined in Figure 11.

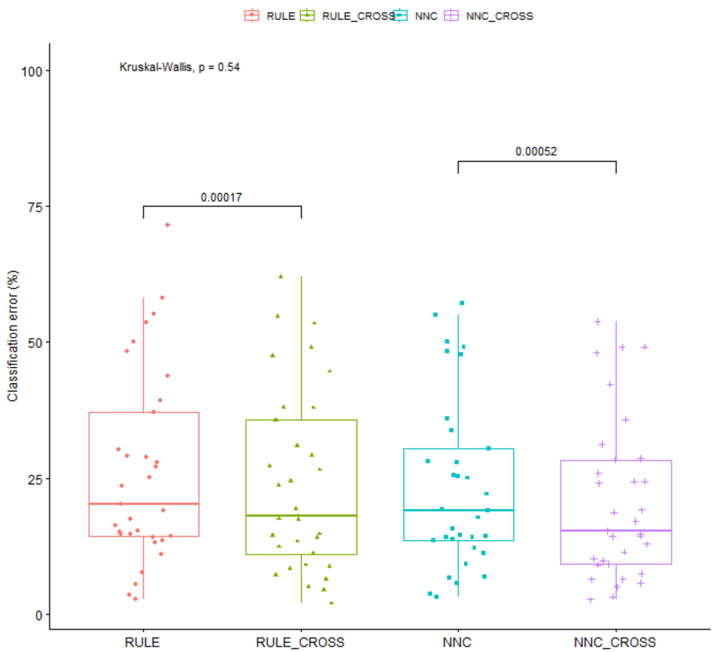

Figure 11. Statistical comparison between the improved machine learning methods and the original methods for the classification datasets.

Moreover, an additional test was executed in order to estimate the effectiveness of the new crossover rate parameter denoted as p_{cr}. In this experiment the rule construction machine learning model was applied on the classification datasets using a series of values for p_{cr} and the results are shown in Table 4.

Table 4. The effect of different values of p_{cr} to the RULE model with application on the classification datasets.

DATASET	**RULE**	$p_{cr} = 0.025$	$p_{cr} = 0.05$	$p_{cr} = 0.075$
APPENDICITIS	14.70%	15.80%	14.80%	15.10%
AUSTRALIAN	14.27%	13.96%	14.46%	14.03%
BALANCE	28.79%	20.18%	17.47%	18.07%
CIRCULAR	13.25%	11.00%	9.12%	9.78%
CLEVELAND	48.24%	48.24%	47.52%	46.07%
DERMATOLOGY	43.77%	38.60%	38.00%	36.00%
ECOLI	55.18%	52.49%	53.48%	48.83%
FERT	17.40%	16.70%	17.50%	18.50%
HABERMAN	27.03%	27.57%	26.53%	26.87%
HAYES-ROTH	39.39%	35.69%	38.08%	36.77%
HEART	20.30%	20.48%	19.41%	20.37%
HEARTATTACK	23.63%	22.83%	23.70%	22.53%
HOUSEVOTES	3.48%	3.48%	4.51%	3.13%
GLASS	58.10%	55.62%	54.81%	52.76%
IONOSPHERE	15.06%	15.14%	14.14%	14.14%
LIVERDISORDER	37.09%	34.79%	35.68%	33.50%
MAMMOGRAPHIC	19.00%	18.34%	18.10%	17.90%
PARKINSONS	13.47%	13.95%	13.37%	13.21%
PIMA	27.85%	27.80%	27.30%	27.84%
POPFAILURES	5.44%	5.33%	5.02%	5.32%
REGIONS2	29.13%	28.82%	29.26%	28.00%
SAHEART	30.20%	30.00%	31.00%	30.18%
SEGMENT	71.51%	67.36%	61.99%	63.91%
SPIRAL	50.06%	50.42%	49.08%	49.60%
STUDENT	11.08%	7.50%	7.23%	6.07%
TRANSFUSION	25.19%	24.20%	24.46%	24.68%
WDBC	7.66%	5.79%	6.43%	6.41%
WINE	15.35%	15.47%	12.47%	13.59%
Z_F_S	16.40%	11.63%	8.77%	9.10%
Z_O_N_F_S	53.64%	47.14%	44.60%	44.04%
ZO_NF_S	14.10%	10.50%	8.39%	8.42%
ZONF_S	2.76%	2.64%	2.06%	2.14%
ZOO	14.80%	11.30%	11.10%	8.70%
AVERAGE	**26.28%**	**24.57%**	**23.93%**	**23.50%**

Looking at the table of results, one can see a significant decrease in the average classification error when the application rate of the genetic operator increases from 2.5% to 5%. However, the rate of reduction of the average error decreases significantly when the application rate increases to 7.5%. This finding reinforces the idea of implementing the new genetic operator at a rate of 5%.

Another experiment was performed in order to estimate the importance of the parameter N_{cr}, which controls the number of chromosomes participating in the new crossover operator. In this experiment the neural network construction method was applied on the classification datasets using different values for the parameter N_{cr} while the parameter p_{cr} was fixed to 2.5%. The outcomes obtained from this experiment are depicted in Table 5.

Table 5. The effects of the parameter N_{cr} to the NNC machine learning model. The experiments were conducted on the classification datasets. In all experiments the value p_{cr} was set to 0.025.

DATASET	**NNC**	$N_{cr} = 25$	$N_{cr} = 50$	$N_{cr} = 100$
APPENDICITIS	13.70%	14.00%	14.50%	14.70%
AUSTRALIAN	14.51%	14.46%	14.13%	13.97%
BALANCE	22.11%	22.29%	17.76%	18.05%
CIRCULAR	13.64%	11.90%	9.38%	8.46%
CLEVELAND	50.10%	49.69%	48.90%	49.17%
DERMATOLOGY	25.06%	20.51%	18.20%	16.29%
ECOLI	47.82%	47.79%	47.39%	47.52%
FERT	19.00%	18.70%	19.20%	18.70%
HABERMAN	28.03%	28.27%	28.43%	26.70%
HAYES-ROTH	35.93%	31.54%	27.77%	27.69%
HEART	15.78%	15.07%	16.00%	14.67%
HEARTATTACK	19.33%	20.13%	19.73%	18.50%
HOUSEVOTES	3.65%	3.30%	3.26%	3.13%
GLASS	57.10%	55.38%	54.62%	54.29%
IONOSPHERE	11.12%	10.63%	10.71%	9.89%
LIVERDISORDER	33.71%	32.03%	32.53%	31.12%
MAMMOGRAPHIC	17.78%	17.72%	17.64%	17.12%
PARKINSONS	12.21%	12.53%	12.79%	11.58%
PIMA	27.99%	27.26%	27.68%	26.09%
POPFAILURES	6.74%	6.33%	6.91%	6.35%
REGIONS2	25.52%	26.20%	25.47%	24.82%
SAHEART	30.52%	30.61%	29.81%	29.58%
SEGMENT	54.99%	53.07%	49.24%	42.90%
SPIRAL	48.39%	48.08%	48.20%	48.34%
STUDENT	5.78%	5.40%	5.20%	4.10%
TRANSFUSION	25.34%	25.26%	24.80%	24.47%
WDBC	6.95%	6.82%	7.39%	6.59%
WINE	14.35%	11.82%	11.77%	9.88%
Z_F_S	14.17%	12.60%	13.50%	9.98%
Z_O_N_F_S	49.18%	48.20%	46.24%	44.73%
ZO_NF_S	14.14%	12.72%	12.18%	10.42%
ZONF_S	3.14%	3.18%	2.82%	2.58%
ZOO	9.20%	8.20%	8.10%	7.50%
AVERAGE	**23.54%**	**22.78%**	**22.19%**	**21.21%**

The lowest average classification error is observed for $N_{cr} = 100$, however, no major changes are observed in the classification errors as the parameter increases. Furthermore, is expected the average execution time to increase as the value N_{cr} increases and this is demonstrated in Figure 12, where the average execution time for the neural network construction method is plotted with respect to the N_{cr} .

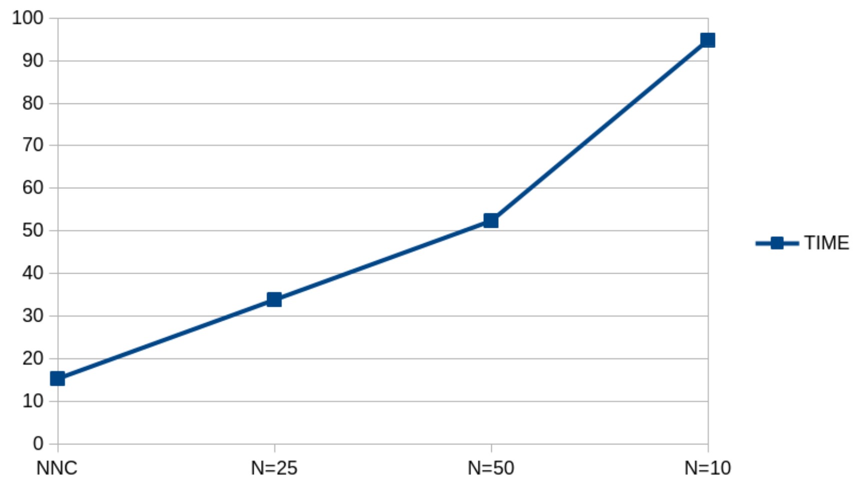

Figure 12. Average execution time for the NNC machine learning model using different values of the N_{cr} value.

The average execution time increases dramatically as the critical parameter N_{cr} increases, something that is expected since the crossings increase significantly with the increase of this parameter, as well as the evaluation of the fitness function. This dramatic increase in required execution time can be significantly reduced by using modern parallel libraries. These programming techniques may include the application of the Message Passing Interface (MPI) library [97] or the incorporation of the OpenMP library [98].

4. Conclusions

A new genetic operator for tasks based on Grammatical Evolution is introduced in this article. This operator is applied to randomly selected chromosomes of the genetic population. On each application, a group of randomly selected chromosomes is formulated for every chromosome and one - point crossover is executed between each member of the group and the selected chromosome, aiming to reduce the associated fitness value. In order to measure the effectiveness of the new operator, it was applied with success in two machine learning models from the recent bibliography that utilize the Grammatical Evolution method:

- A rule construction method, that constructs rules in language similar to the C programming language for data classification or regression problems.
- A method that constructs artificial neural networks.

The methods were applied on a wide series of datasets in the recent literature. In the vast majority of cases, the application of the new genetic operator resulted in a drastic reduction of the corresponding classification or data fitting error. Furthermore, to assess the effect of changing the values of the critical parameters of the genetic operator on the performance of the machine learning methods, more experiments were conducted in which these critical parameters were changed over a wide range of values. Boosting these values improves the performance of machine learning methods by applying the new genetic operator, but up to a point. In addition, an increase in the chromosomes involved in the genetic operator, has a significant increase in the required execution time, as was also seen in the performed experiments. However, with the use of recent techniques that can utilize modern parallel computing structures, this additional time can be significantly reduced.

Improvements of the proposed operator in future studies may include the application of the new crossover in other machine learning methods based on Grammatical Evolution, a parallel implementation of the operator or even the usage of this operator in other tasks involving Genetic Algorithms.

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. J.H. Holland, Genetic algorithms. Scientific american **267**, pp. 66-73, 1992.
2. N. Yusup, A. M. Zain, S. Z. M. Hashim, Evolutionary techniques in optimizing machining parameters: Review and recent applications (2007–2011), Expert Systems with Applications **39.10**, pp. 9909-9927, 2012.
3. J. Stender, Parallel Genetic Algorithms:Theory & Applications. Edition: IOS Press, 1993.
4. D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing Company, Reading, Massachussets, 1989.
5. Z. Michaelewicz, Genetic Algorithms + Data Structures = Evolution Programs. Springer - Verlag, Berlin, 1996.
6. Y.H. Santana, R.M. Alonso, G.G. Nieto, L. Martens, W. Joseph, D. Plets, Indoor genetic algorithm-based 5G network planning using a machine learning model for path loss estimation, Appl. Sci. **12**, 3923. 2022.
7. X. Liu, D. Jiang, B. Tao, G. Jiang, Y. Sun, J. Kong, B. Chen, Genetic algorithm-based trajectory optimization for digital twin robots, Front. Bioeng. Biotechnol **9**, 793782, 2022.
8. K. Nonoyama, Z.Liu, T. Fujiwara, M.M. Alam, T. Nishi, Energy-efficient robot configuration and motion planning using genetic algorithm and particle swarm optimization, Energies **15**, 2074, 2022.
9. K. Liu, B. Deng, Q. Shen, J. Yang, Y. Li, Optimization based on genetic algorithms on energy conservation potential of a high speed SI engine fueled with butanol–gasoline blends, Energy Rep. **8**, pp. 69–80, 2022.
10. G. Zhou, S. Zhu, S. Luo, Location optimization of electric vehicle charging stations: Based on cost model and genetic algorithm, Energy **247**, 123437, 2022.
11. Doewes, R.I.; Nair, R.; Sharma, T. Diagnosis of COVID-19 through blood sample using ensemble genetic algorithms and machine learning classifier. World J. Eng. 2022, 19, 175–182.
12. Choudhury, S.; Rana, M.; Chakraborty, A.; Majumder, S.; Roy, S.; RoyChowdhury, A.; Datta, S. Design of patient specific basal dental implant using Finite Element method and Artificial Neural Network technique. J. Eng. Med. 2022, 236, 1375–1387.
13. Chen, Q.; Hu, X. Design of intelligent control system for agricultural greenhouses based on adaptive improved genetic algorithm for multi-energy supply system. Energy Rep. 2022, 8, 12126–12138.
14. I.G. Tsoulos, Learning Functions and Classes Using Rules , AI **3**, pp. 751-763, 2022.
15. M. O'Neill, C. Ryan, Grammatical evolution, IEEE Trans. Evol. Comput. **5**,pp. 349–358, 2001.
16. 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.
17. 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.
18. 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.
19. A. Brabazon, M. O'Neill, Credit classification using grammatical evolution, Informatica **30.3**, 2006.
20. S. Şen, J.A. Clark. A grammatical evolution approach to intrusion detection on mobile ad hoc networks, In: Proceedings of the second ACM conference on Wireless network security, 2009.
21. I.G. Tsoulos, I.E. Lagaris, Solving differential equations with genetic programming, Genet Program Evolvable Mach **7**, pp. 33–54, 2006.
22. L. Chen, C.H. Tan, S.J. Kao, T.S. Wang, Improvement of remote monitoring on water quality in a subtropical reservoir by incorporating grammatical evolution with parallel genetic algorithms into satellite imagery, Water Research **42**, pp. 296-306, 2008.
23. J. Tavares, F.B. Pereira, Automatic Design of Ant Algorithms with Grammatical Evolution. In: Moraglio, A., Silva, S., Krawiec, K., Machado, P., Cotta, C. (eds) Genetic Programming. EuroGP 2012. Lecture Notes in Computer Science, vol 7244. Springer, Berlin, Heidelberg, 2012.

24. C. Ryan, M. O'Neill, J.J. Collins, Grammatical evolution: Solving trigonometric identities, proceedings of Mendel. Vol. 98. 1998.
25. 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.
26. Lídio Mauro Limade Campo, R. Célio Limã Oliveira,Mauro Roisenberg, Optimization of neural networks through grammatical evolution and a genetic algorithm, Expert Systems with Applications **56**, pp. 368-384, 2016.
27. K. Soltanian, A. Ebnenasir, M. Afsharchi, Modular Grammatical Evolution for the Generation of Artificial Neural Networks, Evolutionary Computation **30**, pp 291–327, 2022.
28. I. Dempsey, M.O' Neill, A. Brabazon, Constant creation in grammatical evolution, International Journal of Innovative Computing and Applications **1** , pp 23–38, 2007.
29. 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.
30. 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.
31. 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.
32. N. R. Sabar, M. Ayob, G. Kendall, R. Qu, Grammatical Evolution Hyper-Heuristic for Combinatorial Optimization Problems, IEEE Transactions on Evolutionary Computation **17**, pp. 840-861, 2013.
33. 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.
34. P.J. Pereira, P. Cortez, R. Mendes, Multi-objective Grammatical Evolution of Decision Trees for Mobile Marketing user conversion prediction, Expert Systems with Applications **168**, 114287, 2021.
35. F. Castejón, E.J. Carmona, Automatic design of analog electronic circuits using grammatical evolution, Applied Soft Computing **62**, pp. 1003-1018, 2018.
36. L. Araujo, J. Martinez-Romo, A. Duque, Discovering taxonomies in Wikipedia by means of grammatical evolution. Soft Comput **22**, pp. 2907–2919, 2018.
37. C. Martín, D. Quintana, P. Isasi, Grammatical Evolution-based ensembles for algorithmic trading, Applied Soft Computing **84**, 105713, 2019.
38. Moore, J.H., Sipper, M. (2018). Grammatical Evolution Strategies for Bioinformatics and Systems Genomics. In: Ryan, C., O'Neill, M., Collins, J. (eds) Handbook of Grammatical Evolution. Springer, Cham. https://doi.org/10.1007/978-3-319-78717-6_16
39. Hidalgo, J. I., Colmenar, J. M., Risco-Martin, J. L., Cuesta-Infante, A., Maqueda, E., Botella, M., & Rubio, J. A. (2014). Modeling glycemia in humans by means of grammatical evolution. Applied Soft Computing, 20, 40-53.
40. A. Bartoli, M. Castelli, E. Medvet, Weighted Hierarchical Grammatical Evolution, IEEE Transactions on Cybernetics **50**, pp. 476-488, 2020.
41. N. Lourenço, F.B. Pereira, E, Costa, Unveiling the properties of structured grammatical evolution, Genetic Programming and Evolvable Machines **17** , pp. 251-289, 2016.
42. N. Lourenço, F. Assunção, F.B. Pereira, E. Costa, P. Machado, Structured grammatical evolution: a dynamic approach, Handbook of grammatical evolution, pp. 137-161, 2018.
43. O'Neill, M., Swafford, J. M., McDermott, J., Byrne, J., Brabazon, A., Shotton, E., Hemberg, M. (2009, July). Shape grammars and grammatical evolution for evolutionary design. In Proceedings of the 11th Annual conference on Genetic and evolutionary computation (pp. 1035-1042).
44. M. O'Neill, A. Brabazon, M. Nicolau, S.M. Garraghy, P. Keenan, πGrammatical Evolution. In: Deb, K. (eds) Genetic and Evolutionary Computation – GECCO 2004. GECCO 2004. Lecture Notes in Computer Science, vol 3103. Springer, Berlin, Heidelberg, 2004.
45. Riccardo Poli, James Kennedy kennedy, Tim Blackwell, Particle swarm optimization An Overview, Swarm Intelligence **1**, pp 33-57, 2007.
46. M. O'Neill, A. Brabazon, Grammatical swarm: The generation of programs by social programming. Natural Computing **5**, pp. 443-462, 2006.
47. E. Ferrante, E. Duéñez-Guzmán, A.E. Turgut, T. Wenseleers, GESwarm: Grammatical evolution for the automatic synthesis of collective behaviors in swarm robotics. In Proceedings of the 15th annual conference on Genetic and evolutionary computation, pp. 17-24, 2013.
48. J. Mégane, N. Lourenço, P. Machado, Probabilistic Grammatical Evolution. In: Hu, T., Lourenço, N., Medvet, E. (eds) Genetic Programming. EuroGP 2021. Lecture Notes in Computer Science, vol 12691. Springer, Cham, 2021.
49. Tan, Y., Zhu, Y.: Fireworks algorithm for optimization. In: Tan, Y. et al. (eds.) ICSI 2010, Part I, LNCS, vol. 6145, pp. 355–364. Springer, Heidelberg (2010).
50. Tapas Si (2016). Grammatical Evolution Using Fireworks Algorithm. In: Pant, M., Deep, K., Bansal, J., Nagar, A., Das, K. (eds) Proceedings of Fifth International Conference on Soft Computing for Problem Solving. Advances in Intelligent Systems and Computing, vol 436. Springer, Singapore. https://doi.org/10.1007/978-981-10-0448-3_4

51. I. Contreras, R. Calm, M.A. Sainz, P. Herrero, J. Vehi, Combining Grammatical Evolution with Modal Interval Analysis: An Application to Solve Problems with Uncertainty, Mathematics **9**, 631, 2021.
52. O. Popelka, P. Osmera, Parallel Grammatical Evolution for Circuit Optimization. In: Hornby, G.S., Sekanina, L., Haddow, P.C. (eds) Evolvable Systems: From Biology to Hardware. ICES 2008. Lecture Notes in Computer Science, vol 5216. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-85857-7_40
53. P. Ošmera, Two level parallel grammatical evolution, Advances in Computational Algorithms and Data Analysis pp. 509-525, 2009.
54. A. Ortega, M. de la Cruz, M. Alfonseca, Christiansen Grammar Evolution: Grammatical Evolution With Semantics, IEEE Transactions on Evolutionary Computation **11**, pp. 77-90, Feb. 2007.
55. M. O'Neill, E. Hemberg, C. Gilligan, E. Bartley, J. McDermott, A. Brabazon, GEVA: grammatical evolution in Java. ACM SIGEVOlution **3**, pp. 17-22, 2008.
56. L. Georgiou, W.J. Teahan, jGE-A java implementation of grammatical evolution. In 10th WSEAS International Conference on Systems, Athens, Greece, 2006.
57. F. Noorian, A.M. de Silva, P.H.W. Leong, gramEvol: Grammatical Evolution in R, Journal of Statistical Software **71**, pp. 1–26, 2016.
58. A. de Lima, S. Carvalho, D.M. Dias, E. Naredo, J.P. Sullivan, C. Ryan, GRAPE: Grammatical Algorithms in Python for Evolution. Signals **3**, pp. 642-663, 2022.
59. M.A. Raja, C. Ryan, GELAB - A Matlab Toolbox for Grammatical Evolution. In: Yin, H., Camacho, D., Novais, P., Tallón-Ballesteros, A. (eds) Intelligent Data Engineering and Automated Learning – IDEAL 2018. IDEAL 2018. Lecture Notes in Computer Science(), vol 11315, 2018. Springer, Cham. https://doi.org/10.1007/978-3-030-03496-2_22
60. N.Anastasopoulos, I.G. Tsoulos, A. Tzallas, GenClass: A parallel tool for data classification based on Grammatical Evolution, SoftwareX **16**, 100830, 2021.
61. I.G. Tsoulos, QFC: A Parallel Software Tool for Feature Construction, Based on Grammatical Evolution, Algorithms **15**, 295, 2022.
62. I.G. Tsoulos, Learning Functions and Classes Using Rules, AI. **3**, pp.751-763, 2022.
63. I.G. Tsoulos, D. Gavrilis, E. Glavas, Neural network construction and training using grammatical evolution. Neurocomputing **72**, pp. 269-277, 2008.
64. M. Kelly, R. Longjohn, K. Nottingham, The UCI Machine Learning Repository. 2023. Available online: https://archive.ics.uci.edu (accessed on 18 February 2024).
65. 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.
66. 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.
67. J.R. Quinlan, Simplifying Decision Trees. International Journal of Man-Machine Studies **27**, pp. 221-234, 1987.
68. T. Shultz, D. Mareschal, W. Schmidt, Modeling Cognitive Development on Balance Scale Phenomena, Machine Learning **16**, pp. 59-88, 1994.
69. 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.
70. R. Setiono , W.K. Leow, FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks, Applied Intelligence **12**, pp. 15-25, 2000.
71. 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.
72. 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.
73. 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.
74. I. Kononenko, E. Šimec, M. Robnik-Šikonja, Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF, Applied Intelligence **7**, pp. 39–55, 1997
75. 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.
76. J. Garcke, M. Griebel, Classification with sparse grids using simplicial basis functions, Intell. Data Anal. **6**, pp. 483-502, 2002.
77. 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.
78. 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.
79. 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.
80. 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.

81. 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.
82. T. Hastie, R. Tibshirani, Non-parametric logistic and proportional odds regression, JRSS-C (Applied Statistics) **36**, pp. 260–276, 1987.
83. M. Dash, H. Liu, P. Scheuermann, K. L. Tan, Fast hierarchical clustering and its validation, Data & Knowledge Engineering **44**, pp 109–138, 2003.
84. 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.
85. W.H. Wolberg, O.L. Mangasarian, Multisurface method of pattern separation for medical diagnosis applied to breast cytology, Proc Natl Acad Sci U S A. **87**, pp. 9193–9196, 1990.
86. 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.
87. P. Zhong, M. Fukushima, Regularized nonsmooth Newton method for multi-class support vector machines, Optimization Methods and Software **22**, pp. 225-236, 2007.
88. 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, Phys. Rev. E **64**, pp. 1-8, 2001.
89. M. Koivisto, K. Sood, Exact Bayesian Structure Discovery in Bayesian Networks, The Journal of Machine Learning Research **5**, pp. 549–573, 2004.
90. 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 48; Sea Fisheries Division, Department of Primary Industry and Fisheries: Orange, NSW, Australia, 1994.
91. T.F. Brooks, D.S. Pope, A.M. Marcolini, Airfoil self-noise and prediction. Technical report, NASA RP-1218, July 1989.
92. J.S. Simonoff, Smooting Methods in Statistics, Springer - Verlag, 1996.
93. I.Cheng Yeh, Modeling of strength of high performance concrete using artificial neural networks, Cement and Concrete Research. **28**, pp. 1797-1808, 1998.
94. D. Harrison and D.L. Rubinfeld, Hedonic prices and the demand for clean ai, J. Environ. Economics & Management **5**, pp. 81-102, 1978.
95. M.J.D Powell, A Tolerant Algorithm for Linearly Constrained Optimization Calculations, Mathematical Programming **45**, pp. 547-566, 1989.
96. I.G. Tsoulos, Modifications of real code genetic algorithm for global optimization, Applied Mathematics and Computation **203**, pp. 598-607, 2008.
97. Gropp, W.; Lusk, E.; Doss, N.; Skjellum, A. A high-performance, portable implementation of the MPI message passing interface standard. Parallel Comput. 1996, 22, 789–828.
98. Chandra, R. Parallel Programming in OpenMP; Morgan Kaufmann: Cambridge, MA, USA, 2001.